University of Bath

PHD

Automatic digitisation and analysis of facial topography by using a biostereometric structured light system

Shokouhi, Shahriar B.

Award date:
1999

Awarding institution:
University of Bath

Link to publication

Alternative formats
If you require this document in an alternative format, please contact:
openaccess@bath.ac.uk

Copyright of this thesis rests with the author. Access is subject to the above licence, if given. If no licence is specified above, original content in this thesis is licensed under the terms of the Creative Commons Attribution-NonCommercial 4.0 International (CC BY-NC-ND 4.0) Licence (https://creativecommons.org/licenses/by-nc-nd/4.0/). Any third-party copyright material present remains the property of its respective owner(s) and is licensed under its existing terms.

Take down policy
If you consider content within Bath's Research Portal to be in breach of UK law, please contact: openaccess@bath.ac.uk with the details. Your claim will be investigated and, where appropriate, the item will be removed from public view as soon as possible.

Download date: 05. May. 2023

Automatic Digitisation and Analysis of Facial Topography by Using a Biostereometric Structured Light System

Submitted by Shahriar B. Shokouhi

for the degree of PhD

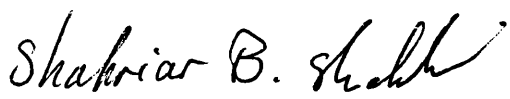
University of Bath

1999

COPYRIGHT

Attention is drawn to the fact that copyright of this thesis rests with its author. This copy of the thesis has been supplied on condition that anyone who consults it is understood to recognise that its copyright rests with its author and that no quotation from the thesis and no information derived from it may be published without the prior written consent of the author.

This thesis may be made available for consultation within the University Library and may be photocopied or lent to other libraries for the purposes of consultation.



UMI Number: U114559

All rights reserved

INFORMATION TO ALL USERS
The quality of this reproduction is dependent upon the quality of the copy submitted.

In the unlikely event that the author did not send a complete manuscript
and there are missing pages, these will be noted. Also, if material had to be removed,
a note will indicate the deletion.

UMI U114559
Published by ProQuest LLC 2013. Copyright in the Dissertation held by the Author.
Microform Edition © ProQuest LLC.
All rights reserved. This work is protected against
unauthorized copying under Title 17, United States Code.

ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

UNIVERSITY OF BATH
LIBRARY
- 6 JUL 1999
PHD

Abstract

This thesis describes a new implementation of a non-invasive biostereometric structured light system capable of measuring the shape of a face. A monochrome parallel 2-D structured light technique is used by projecting a special pattern for each side of a face. The system has been used successfully to measure the human face and static objects with a millimetre accuracy.

The work was motivated by the need to acquire information from a face for a clinical environment so that the results presented are for measurements of deformed faces. Primary objectives of this work are to minimise the amount of manual intervention required and to use a simple acquisition technique, so that the system can be used by clinicians.

In clinical use the patient's head is fixed in a head restraint and the optical set-up is adjusted by using a preview facility to obtain an optimum image. Then the face is illuminated by using a brick pattern. The images from two cameras, one for each side of the face, are captured by high resolution hardware and transferred to the computer memory for on line image processing.

The work covers the entire implementation of the optic system, hardware, feature extraction, feature interpretation, calibration, 3-D reconstruction and summary of results on a series of candidate faces. An optical calibration technique is used which includes both camera and projector calibration. The classical feature extraction algorithms are used for enhancement and edge detection and following them a new line thinning methodology based on pixel following is proposed to produce a single pixel skeleton. The detected features are completed and labelled during the feature interpretation stage that is used for recovering the 3-D data. The novel contour tracing algorithm proposed is based on the difference code and junction sequences. The 3-D reconstructed image from the whole face is presented by a user friendly graphics environment to permit the user to manipulate the facial image.

Acknowledgements

The author wishes to express sincere thanks to his supervisors, Dr. Graham Allen and Prof. R. F. Ormondroyd for their guidance and encouragement through this study . My thanks also go to Mr. Martin for his useful advice and kind consideration and also Prof. Watson as the head of the Department of Electronic and Electrical Engineering.

I wish to express my appreciation to Dr. Evans for his useful discussions of the field of this research and to the technician team in the department led by Mr. J. Dowse. Also I wish to thank Mr. Chris Hubbal for his great help for developing the graphics environment, and all the people who have donated their time for this project.

I wish to thank my sponsors, the Ministry of Culture and Higher Education of Iran and Iran university of Science and Technology, for paying my tuition fees and living costs during this study. Last but not least, I like to thank my wife, Nastaran, for her kind co-operation.

Table of Contents

List of Notations

C_1	Left side camera
C_2	Right side camera
P	Projector
S	Set pixel or current pixel
θ	The angle between the camera and projector
d_c	Distance of the camera from the base plane
d_{cp}	Distance between the camera and projector
d_p	Distance of the camera and projector
d_g	Distance between two vertical lines in the slide
d_B	Distance between two black gaps on the slide
d_W	Slit width on the slide
d_o	Distance between two projected stripes on the base plane
d_e	Distance of the plane e from origin
f_c	Camera focal length
f_p	Projector focal length
(x_i, y_i) or (u_i, v_i)	Image co-ordinates in the camera recording plane
$(x`, y`, z`)$	CRP co-ordinate system
(x, y, z)	World co-ordinate system positioned on the patient positioning
(x_u, y_u)	Perfect image co-ordinate in the camera recording plane
(x_d, y_d)	Distorted image co-ordinate in the camera recording plane
(x_o, y_o)	Co-ordinate of the overlapping point
(x_s, y_s)	Image co-ordinate in the computer screen
(x_e, y_e)	Co-ordinates of the end points
(x_t, y_t)	Co-ordinates of the junctions
(x_l, y_l)	Co-ordinates of the start junctions for labelling
(x_g, y_g)	Slide co-ordinate system
$(x`_g, y`_g)$	Corresponding slide co-ordinate system on the screen
(p_s, q_s)	Corresponding junction co-ordinate on the screen
$p(i, j)$	The current pixel co-ordinate on the screen
$\sigma(x, y)$	Grey level image standard deviation
$m(x, y)$	Grey level image mean value
M	Global mean
$\mathbf{C}(c_{11}, c_{12}, ..., c_{34})$	Camera calibration matrix
$\mathbf{P}(p_{11}, p_{12}, ..., p_{34})$	Projector calibration matrix
$\mathbf{R}$	Rotation matrix
$\mathbf{T}$	Translation vector
β_c	Camera amplification factor
β_p	Projector amplification factor
s	Scaling factor
φ_p	Projector optic path
κ_i	Distortion coefficient
m	median
ε	Bresenham difference value
e	Vertical line index number
e_d	End point direction

$\underline{n}$	Plane normal
E_B	Boundary end points
T_B	Boundary junctions
N_B	Boundary network
E_L	Loop end points
T_L	Loop junctions
N_L	loop network
a_t	Junction attribute
S_k	Set of contour pixels of a binary image
C_k	The direction chain code set of contour k in an image
c_i	The direction code of pixel i in a contour
X_k	x co-ordinate set of contour k in an image
Y_k	y co-ordinate set of contour k in an image
d_i	The difference code between c_i and c_{i+1}
d_{ij}	The difference code between pixel i and j of a line
$c`_i$	The inverse direction code of pixel i in a contour
$d`_i$	The difference code between $c`_i$ and $c`_{i+1}$
T_{ij}	Direct junction with co-ordinate of (i, j)
$T`_{ij}$	Corresponding junction with co-ordinate of (i, j)
L_{ij}	A closed loop indexed with (i, j)
E_i	End point with number i
E_o	Overlapped point
n_e	Number of end points inside the loops
n_t	Number of junctions inside the loops
n_d	Number of disconnected loops
n_b	Number of boundary loops
X_{right}	Right border pixel for 7 by 7 window
X_{left}	Left border pixel for 7 by 7 window
X_{top}	Top border pixel for 7 by 7 window
X_{bottom}	Bottom border pixel for 7 by 7 window
$X`_{left}$	Left border pixel for 3 by 3 window
$X`_{top}$	Top border pixel for 3 by 3 window
$X`_{bottom}$	Bottom border pixel for 3 by 3 window
c_c	Circle of confusion
h	hyperfocal distance
D	Apparture
N_e	Bellows factor
μ	The locus of the spline curve
W	Bernstein blending function

List of Abbreviations

1-D	One Dimensional
2-D	Two Dimensional
3-D	Three Dimensional
ADC	Analogue to Digital Converter
ADC_EN	Analogue to Digital Converter Enable Signal
AGC	Automatic Gain Control
APIN	One pixel image data (Odd)
BEP	Bottom End Point
BPIN	One pixel image data (Even)
BPORCH	Back Porch
BSLS	Biostereometric Structured Light System
CAD	Computer Aided Design
CCD	Charged Coupled Device
CCTV	Close Circuit TV
CJLA	Corresponding Junction Labelled Array
CLK	Clock
CPIN	8 bit control data
CRP	Camera Recording Plane
CS	Chip Select
CSYNCH	Composite Synchronisation Signal
DAC	Digital to Analogue Converter
DCS	Difference Code Sequence
DJLA	Direct Junction Labelled Array
DLT	Direct Linear Transform
DRAM	Dynamic Random Access Memory
EOL	End of Line
EOF	End of Frame
EP	End Point
EPROM	Erasable Programmable Read Only Memory
GCS	Global Co-ordinate System
IDC	Inverse Directional Code
I/O	Input Output
ISIS	Integrated Shape Imaging System
JCP	Junction Centre Point
JLA	Junction Labelled Array
JS	Junction Sequence
LA	Labelled Array
LEP	Left End Point
LSB	Least Significant Bit
LSN	Least Significant Nibble
MSB	Most Significant Bit
MSE	Mean Square Equation
MSN	Most Significant Nibble
OE	Output Enable
PC	IBM compatible Personal Computer

PCB	Printed Circuit Board
PLD	Programmable Logic Device
PPI	Programmable Peripheral Interface
RAM	Random Access Memory
RAM_ACC	RAM Access Signal
REP	Right End Point
RES_LIN_C	Reset Line Counter
RES_PIX_C	Reset Pixel Counter
SC	Slide Controller
SRAM	Static Random Access Memory
SVGA	Super Video Graphics Adapter
TEP	Top End Point
VAS	Video Active Splitter
VSYNCH	Vertical Synchronisation Signal

List of Figures

List of Tables

Introduction

Three dimensional (3-D) images have recently received wide attention in applications involving medical treatment. Most current 3-D imaging methods focus on the internal organs of the body. However, some specialisations such as plastic surgery, rehabilitation, dental surgery and orthodontics, make use of the surface contours of the body for diagnosis, treatment planning and prediction of postoperative soft-tissue profile. Several techniques are currently available for producing 3-D images of the body surface, such as sterephotogrammetry [1], light stripe scanning systems [2], Moiré topography [3], laser scanning systems [4] and phase measuring profilometry [5]. Most of the systems which implement these techniques are expensive, requiring complex equipment with highly trained operators. All current implementations involve extensive manual intervention and this has proved to be a serious hindrance to the clinical acceptance of 3-D imaging. A primary objective of this work is to minimise the amount of manual intervention required, so that the system can be used by clinicians who do not have special training in the use of the equipment.

Biostereometrics can be defined as the spatial and spatiotemporal analysis of biological form utilising the principles of analytical geometry. The widespread use of biostereometrics has been restricted by the time and cost factors involved in the acquisition of accurate, comprehensive three-dimensional measurements of human soft tissue. Conventional anthropometric methods, based on the use of callipers, offer uncomplicated and practical two dimensional assessment but are inadequate for the task of surface characterisation. The structured light-based optical methods are suitable for the measurement of the surface shape of objects when physical contact is undesirable, especially in the medical field. Usually these techniques rely on the projection of a pattern of light and dark stripes or grids onto the object in question, and the recording of one or more images of it. The reflected light produced is a distorted version of the original pattern. This distortion is directly related to the surface of the subject and so it is possible to calculate the contours of the surface by comparing the distorted image with the expected projection of the original pattern.

Structured light, in which two or more images taken from different view points are analysed to yield spatial information, has been used in biometric studies since the early 1950s [6] and has been successfully applied to facial measurement [7]. Traditional structured light systems required the use of film emulsions and expensive pre-calibrated metric cameras. The last decade saw the development of low-cost and high resolution CCD cameras that provide the advantage of image capture in a format suitable for digital input to computer systems and subsequent automatic analysis.

This thesis describes the implementation of a biostereometric structured light system (BSLS) capable of measuring the shape of a face. This is achieved using two CCD cameras, a commercial slide projector and the high resolution frame store hardware linked to a PC. The images from each side of a face are captured by the cameras and then digitised and stored in the computer memory. This allows the image processing to be performed on-line. Novel algorithms for recovering data from the structured light

images have been developed, and the system as a whole has been taken far beyond the level of the original biostereometric systems proposed by *Lewis et al.* [8] and *Deacon et al.* [9].

Structured light is a general concept and there are a number of ways of exploiting it to obtain 3-D information. Two broad class of the structured light systems have been defined as parallel and serial techniques. Parallel techniques modulate the surface of an object with a structured light pattern, and then recover range data for the entire surface using a single image. But in the serial techniques, the range maps are measured by scanning the surface. Parallel structured light techniques can broadly split into two main categories, those which are stripe-based (1-D pattern), and those which contain coding in a second dimension (2-D pattern), such as spots or a grid. 2-D stripe patterns allow robust matching for the scene to solve the corresponding problem.

The mathematical foundations of structured light have been applied to the new imaging technique to allow the determination of spatial information. The process of 3-D reconstruction consists of applying simple geometry to find the intersection of the camera rays corresponding to the detected features with the appropriate plane of the projected light. The 3-D profile of the object under measurement is obtained with respect to a base plane. The triangulation formula depends on the optical geometry of the system, the optical axes of the acquisition unit, and the intersection of the optical axes with projection line in any point on the subject's surface, and is used to calculate the height information.

The automated system for measuring the facial surface is shown in Figure I.1. The system used a controllable slide projector with enough power to illuminate the surface. Three different type of slides were made by using a lithography-on-glass technique to generate sharp stripes on the subject's face. The pattern of structured light that was chosen is a two dimensional brick pattern. Slide 1 is designed to permit the operator to level and focus the projector and also to adjust the patient's position. For each side of a face one slide is used by considering the profile on the face. Image acquisition in the automated system is done by two CCD cameras with the resolution of 685 × 585 pixel, fitted with 4 mm CCTV lenses. The cameras are connected to a high resolution 512 × 512 based framestore capable of capturing stereo pairs simultaneously.

The clinical use requires a suitable image with regard to equipment set-up and patient positioning. The patient positioning problem is particularly important as it must be remembered that people tire quickly when asked to sit motionless for even the shortest times. The framestore has a hardware preview facility whereby an image can be viewed instantly before grabbing, and the intensity of the camera and patient positioning may be adjusted for optimum results. When the operator decides that the position and expression are acceptable, the framestore 'freezes' the image data, and transfers it to the computer memory for direct display on the high resolution screen. As the image capture time is 1/25 second, image blur due to patient movement is very rare.

The accuracy of the image is sufficient in comparison with other surface imaging techniques. The system is calibrated, and both its accuracy and resolution have been studied in terms of the set-up geometry and the data reconstruction algorithms. A mean-square accuracy of 1 mm has been demonstrated by digitising the human face.

This is set-up and equipment dependent, so a higher resolution camera can increase accuracy and spatial resolution, and different lenses may be used to cover almost arbitrary ranges of subject sizes.

The work covers the entire implementation of the range-finder, including the hardware design, feature extraction, feature interpretation, calibration, 3-D reconstruction and manifesting of the image presented on the computer screen. Each range sample is stored as a three-dimensional point, defining position in a world co-ordinate system.

To aid clarity, this document is divided into eight chapters. *Chapter 1* looks at the anatomy of a face and some medical applications from a clinical view point and if, and how, surface imaging may help. Then the available three dimensional surface imaging techniques will be discussed in this chapter and why the structured light technique is suitable for capturing the facial information.

An anatomical study of the human face has revealed that each of the layers: bone, muscle and skin has a profound influence over the conformation and feature characteristics of the face. Although all human faces have the same physical structure, there are wide variations in form and appearance. The bones determine the overall size and proportions of the head and face, the shape and thickness of muscles are also influential factors. The texture and colour of skin, as well as surface features are important because they are particularly visible attributes of the face.

The common type of the medical applications on a face have also been reviewed. Three dimensional surface imaging systems permit the clinician to evaluate the face. Although the importance of the third dimension in the medical field had been appreciated for a long time, the main difficulty facing surgeons was that of obtaining sufficient three dimensional measurements on the face. Over the last decade, this situation has changed, and a number of measurement systems with an accuracy better than 1 mm are now produced for the facial surface. The 3-D surface imaging techniques which are applicable to the measurement of the human body, and how they may be applied clinically for monitoring and detection are reviewed in this chapter. The structured light method by using a passive light source is selected as a suitable non-invasive method for the human face.

Chapter 2 concentrates on the structured light approach in detail for calculating the height and explains how the system is established. The chapter ends with the description of operating the system and its requirements to achieve an optimum optic system.

The high resolution hardware for capturing the data from the two cameras and transferring it to the computer is explained in *Chapter 3*. The image acquisition system was designed by considering the clinical requirements, performance and reliability. The practical considerations of the hardware are explained in detail.

The use of white light as the structured light source rather than a laser avoids the risk of damage to the eye. However, because of the poorer contrast of the white light stripes, and because the projected white light can not be simultaneously in sharp focus over the entire surface, sophisticated image processing is necessary to produce a

satisfactory image. The goal of the image processing stage is to generate facial contours without discontinuity and having single pixel thickness, by employing two main stages; feature extraction and feature interpretation.

The aim of feature extraction is to detect and enhance the structured light image to yield a representation (feature) which is more useful in our interpretation of that image. Feature extraction is the next stage and *Chapter 4* describes the traditional image processing tools for enhancement and edge detection that are needed together with the novel aspects of the author's work on line thinning. Also it is seen how the difficulty in achieving good features on some parts of a face, and the resulting noise, means that subsequent high level processing must be applied for a robust result.

The image processing stages are applied to the images of each side of a face separately. The image processing algorithms for implementing each side are developed in a general group, left and right group, to make a separate process for each side. The environment for implementing the image processing is shown in Figure I.2. The menu consists of a horizontal bar of key words each of which has a highlighted letter which when pressed will activate that specific function. Below this bar is an area which displays any relevant information about the option selected. As well as offering various image processing techniques the user environment also allows reading and writing of files for storage and retrieval.

With all the low level processing done, *Chapter 5* presents novel algorithms to interpret the structured light image. Here the goal is to transform the two dimensional image into a full 3-D data set. A key step in the 3-D reconstruction using structured light is to solve the grid labelling or matching problem, to find the correct correspondence between the detected grid points in the camera image and the points in the projected grid pattern. The two main stages involved in the procedure are tracing and labelling. The first stage of assigning heights is to label the junctions and lines and then use the calibration matrices to find the actual height. Although this has been solved several times, the difference here is that of correcting the image components before labelling them. The algorithms in this chapter must therefore cope with noisy and broken lines to recover the data set.

The contours of the human face mean that certain parts of the projected grid can not be seen on the captured image (dead shadow areas) and breaks occur in the projected grid itself around the eyes, eyebrows and nose. To remove these defects, software has been designed to fill in the missing information by a tracing strategy. In order to produce a 3-D data set, all of the lines must be perfectly formed. The novel contour tracing algorithm has been proposed to scan the image components by applying the inverse directional coding (IDC). The sequences of the directional code (DCS) and junctions (JS) provide all the necessary information for recovering the image deficiencies.

The final part of this chapter looks at how the 3-D data set may be displayed as a helpful representation for medical applications. A user friendly graphics environment has been developed to produce a wire frame or rendered model of a face as shown in Figure I.3, and by using the mouse facility this model can be interpolated in three dimensions. After reconstruction of the facial sides, the joining algorithm may be used

to reconstruct a complete representation of a face by applying the feature matching technique.

In *Chapter 6* the calibration of the camera and projector is discussed and a suitable technique selected. The camera is calibrated first and the second step is calibrating the projector using the results from the first calibration. The projector model used is linear and is particular to the defined system.

The problems of calibrating metric and non-metric cameras are well known in photogrammetry. More recently, as the emphasis on real time systems has increased, attention has been given to the use and calibration of CCD cameras. Close-range camera calibration generally utilises images of a field of targets of known 3-D co-ordinates. The mathematical method employed in this system was first proposed by *Tsai* [10]. It is a two-stage technique based on the observation that, irrespective of radial lens distortion, the vector from the principal point to a particular image point is parallel to the vector drawn from the extended optical axis to the corresponding object space point. Tsai's model also allows us to consider the scale factor which defines the relationship between the number of samples along a line on the CCD array and the number of samples along a line in the framestore memory.

Chapter 7 presents a summary of results to show the performance of the system and algorithms for capturing and reconstructing the real faces. The images from different types of faces are captured and processed, then the reconstructed images have been used to show the performance of the system.

The study of structured light has been a multi-disciplinary subject, with the participation of physicists, clinicians, mechanical engineers and electrical engineers. The advantage of this is that the problems with structured light have had a broad base from which to find solutions. The work was motivated by the need to acquire information from the face for the clinical requirements in order to permit the manipulation of classified deformities on a face.

The last chapter, *Chapter 8*, is a general conclusions about the implemented system and algorithms along with future works for implementing the system.

This thesis represents a full start to finish treatment of a surface imaging system based on the structured light technique which leads to a three dimensional reconstructed image.

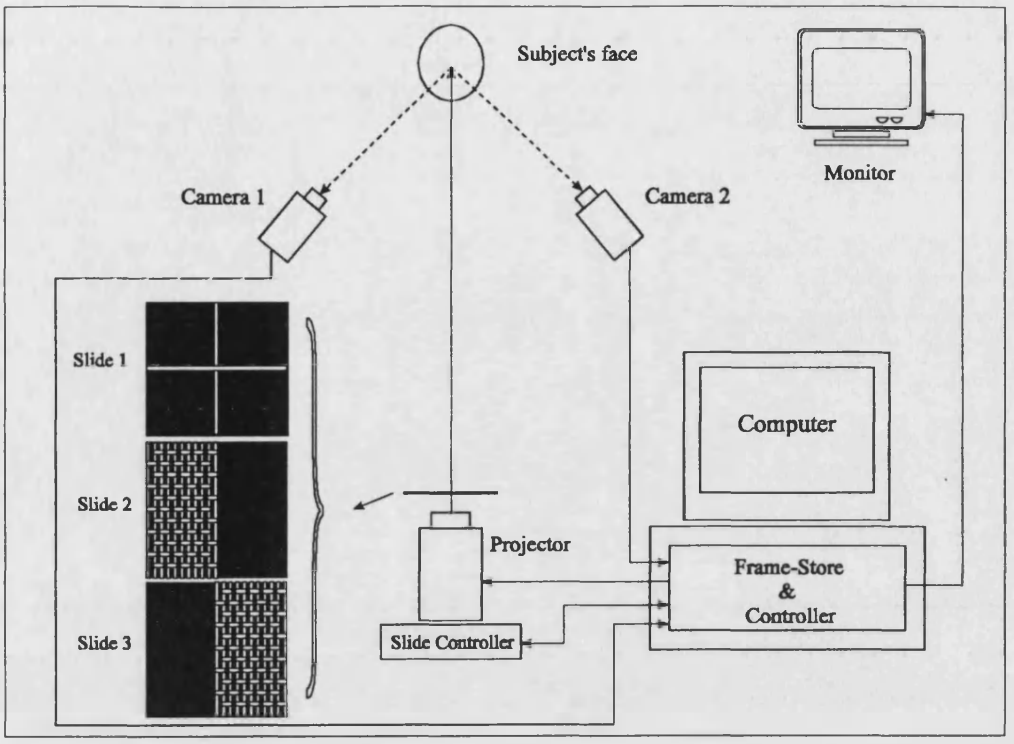
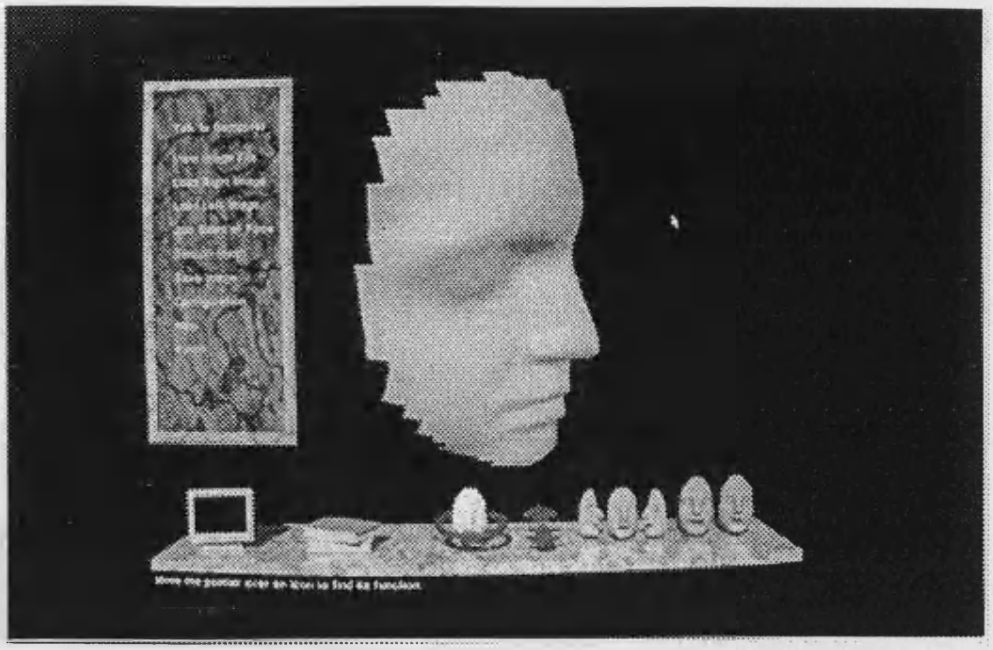
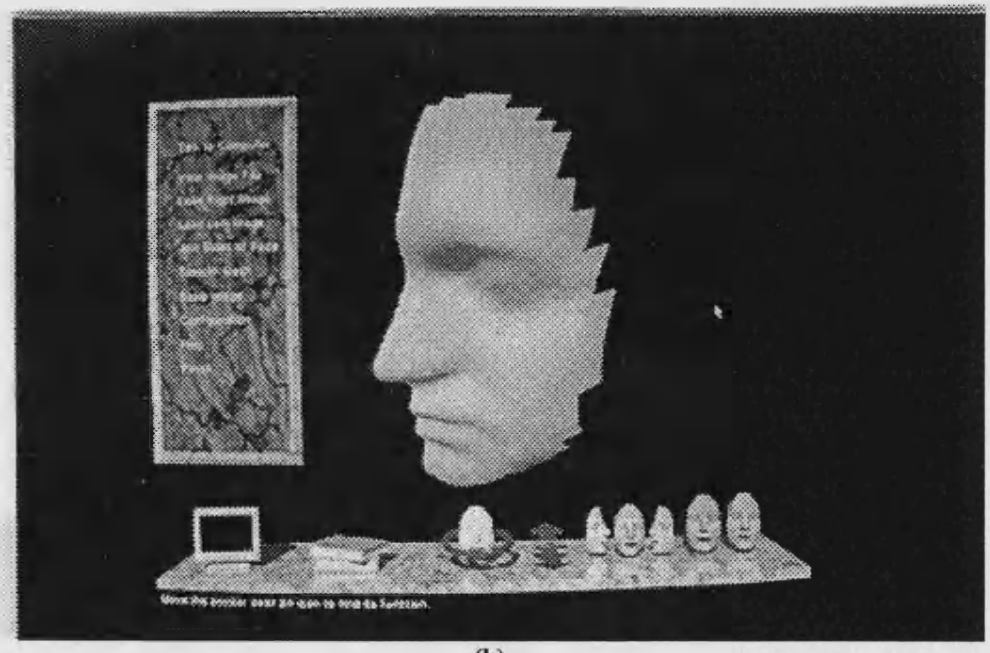


Figure I.1: Configuration of the implemented biostereometric structured light system. Three slides have been used for the system, slide 1 is a reference slide for calibrating the optic set-up and patient positioning, slide 2 is for the left side of the face and slide 3 is for the right side of the face. The hardware includes the frame store and the slide controller for capturing the image and controlling the projector.

Figure I.2: The image processing environment, images from each side of a face are captured and monitored on the screen. The resolution of the display is 1024 by 780 which is a SVGA resolution with 256 colours.



(a)



(b)

Figure I.3: Reconstructed images from the (a) left side and (b) right side of the face on a fully mouse controlled graphics environment.

Chapter 1

Review of the 3-D Surface Imaging Techniques

This chapter starts with a brief explanation of the anatomy and physiology of the face. Bone, muscle, skin tissue and surface features, form the major components of the face [11 , 12]. Also the facial deformities that are to be diagnosed and monitored are described. The conditions are looked at from a medical viewpoint and then engineering solutions to the clinical problems are reviewed.

There are many established techniques for performing three dimensional measurements and some of them have been deployed quite successfully but often, due to factors such as accuracy or complexity, they fall short of the optimum solution. This chapter evaluates these techniques in the light of their suitability for facial imaging applications.

Surface imaging techniques are classified into two broad categories, Active [13] and Passive [14]. Active techniques use a special light source (laser beam), whereas passive techniques do not require a special lighting system (ambient light). Several techniques will be briefly reviewed here: stereophotogrammetry, raster stereophotogrammetry, structured light, moiré, laser scanning and phase measuring profilometry. Some of these methods are complex and require a great deal of equipment such as lasers, multiple projectors and cameras, image sensors and very powerful work stations, while others can be done with limited equipment like a projector, a camera and computer. A useful summary can be found in the reference [15].

1.1 Anatomy of the Human Face

1.1.1 Bone

Major differences in the form of individual faces occur due to several factors including age, race, gender and hereditary reasons [16]. In addition, variability in the size, shape and relative placement of the major bones is responsible for the overall shape and proportions of the head and face. These factors together with deviations in fat and facial tissue are extremely important in determining the appearance of the face.

1.1.2 Muscles

The form of the face is to a large extent dependent on the size, thickness and shape of the major muscles. Muscles lie between the bone and skin. Attachment at the bone is known as the *origin* while the connection into the *fascia* of the skin is called the *insertion.* All facial muscles, with the exception of the *orbicularies oris*, emerge or have origins on the underlying bone and insert into the skin.

1.1.3 Skin

Facial skin is important because it covers a large part of the face. This makes skin a particularly memorable attribute of the face. Skin is significant in determining the appearance of the face. Human skin consists of the *dermis*, which forms a layer of loose irregular connective tissues known as the *hypodermis* or *subcutaneous* surface. This arrangement allows the skin considerable freedom of movement over the muscles and bones that lie underneath.

The visual appearance of facial skin is dependent on several factors including texture, depth and colouration. Texture depends mainly on the glands contained within the skin. The depth of the *dermis* varies over different areas of the face, for example the lips are very thick while the eyelids are thin and delicate. Colour is determined by blood circulation, the presence of pigments and health. Skin-tone also varies greatly from infancy to adulthood, as well as between males and females, and between different races.

1.2 Medical Applications of 3-D facial imaging

Facial deformities can result from accidental damage, or they can be present from birth as congenital abnormalities. These defects may produce an abnormal appearance which can be psychologically damaging, or cause possible added mechanical problems with eating and speech. It is therefore necessary, wherever possible, to attempt to correct these deformities using surgical techniques. 3-D facial imaging is a process which can help to the surgeon to operate on the face very accurately.

1.2.1 Cosmetic surgery

Reconstructive surgery involves the restoration of function to damaged body parts and the rebuilding of normal physical contours when parts of the body, such as the nose, jaw or ears, are missing or disfigured [17]. The large number of automobile accidents in modern times has resulted in many patients requiring reconstructive surgery of the face. Cancer patients who have undergone treatment of the face and neck area may need reconstructive surgery. Huge number of burn patients also need this surgery.

Facial surgery is one of the most complicated types of plastic surgery, requiring artistic as well as technical skills. Plastic surgery today is also often done for cosmetic reasons, to remove defects or to change contours. Among the most common of the cosmetic plastic surgery operations are *rhinoplasty* (remodelling of the nose), *oroplasty* (remodelling of the external ear), *blepharoplasty* (removing excess skin and fatty tissue from eyelids and the eye area), and face-lifting (to remove the signs of ageing).

3-D facial representation gives the surgeon an opportunity to measure the damaged area of the patient's face with a proper scale and allow him to practice any reshaping techniques on the computer image by using software facilities. Expanding the technique to the beauty industry could mean that make up and hairstyles could be applied to a computer representation of the clients face to enable them to see their desired appearance, before any work is commenced.

1.2.2 Dentistry

Orthodontics studies the regulation of the position of teeth in the dental arch [18 , 19]. It deals with the detection, study, prevention, and correction of the condition known as *malocclusion*, in which irregularities in tooth position and jaw relationships lead to deformities of the jaws and face. *Malocclusion* may be hereditary or may be an acquired defect caused by faulty eating habits or early tooth loss.

By applying special devices and appliances to the teeth, sometimes in combination with surgery, a proper occlusion of the teeth can be effected by the orthodontist. A three dimensional map of a patient's teeth or jawbone, stored on computer would be of great help in performing this work. Any movement in the position of the teeth can be

monitored with high precision by using the computer to compare images taken at different times.

1.2.3 Maxilla Facial Surgery

Maxilla facial surgery to be used when one or both jaws are out of alignment [20]. The process which is used to correct this condition is known as bi-maxillary (dual-arch) *orthognathic* surgery, and involves cutting parts of the jawbones and manipulating these blocks of bone in order to give the required facial features. This is done by cutting away wedges of bone (*osteotomies*), and by using bone grafts. Operation of this type are very complicated, since in many cases bones must be moved in three planes. For this reason the planning beforehand is very important. In the theatre the surgeon must know precisely how to move the blocks of bone since the accuracy to which this is done is critical to the success of the operation.

At present this planning is carried out using frontal, lateral and horizontal radiographic views. Various common points are identified on these by means of lead shot, which when strapped to the teeth and other accessible points, appear as bright dots in the X-ray images. From this, the effect of movement of blocks of bone is estimated by a series of steps in which tracings of outlines of blocks are moved about independently, and the surgeon then estimates the required transformation. Combination of 3-D surface images with radiograph images can give actual reference points for surgeons to do this job more easily and effectively.

1.3 3-D Surface Imaging Techniques

1.3.1 Stereophotogrammetry

Stereophotogrammetry uses two cameras to realise three dimensional information in much the same way as binocular vision is used by mammals. Figure 1.1 shows a practical stereophotogrammetry set-up. First of all, correspondence between the two cameras must be established to ensure they are looking at the same feature to be distanced. This can be done with either a prior knowledge of the subject or with reference points such as fibre-optics or light emitting diodes in the field of vision. Assuming both cameras are aligned on point **p**, by knowing distance B and the camera focal lengths then L1 and L2 may be calculated [21].

Knowledge of the precise camera orientations and distortions are essential as errors will be greatly magnified. This information is yielded by the use of control points in the imaging space and least squares estimation techniques such as explained in the direct linear transformation method [22].

This technique has been applied for the facial imaging by several groups. *Waldhaust et al.* (1990) [23] used this technique to measure changes of the human faces for maxilla facial surgery. *Dorffner et al.* (1994) [24] implemented their systems to help the facial surgeon to control the geometric alternations attained by his operation.

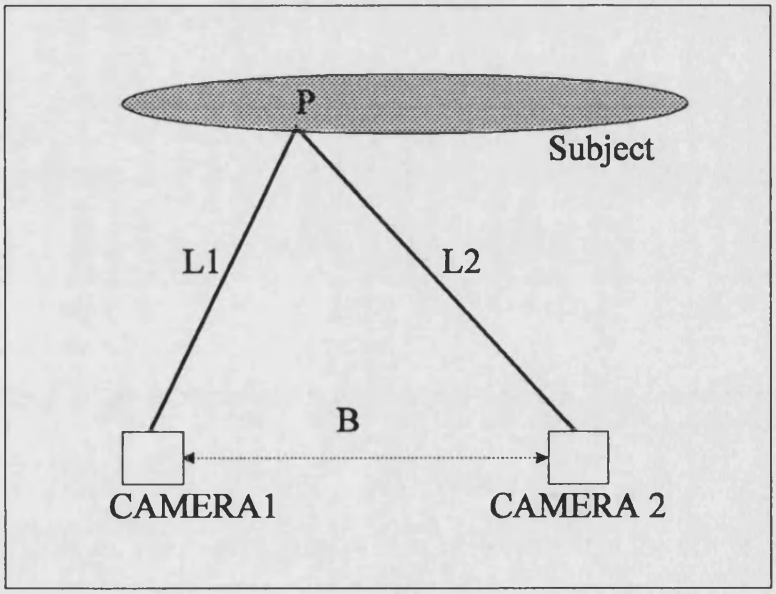


Figure 1.1: Plan of the stereophotogrammetry set-up

1.3.2 Structured Light Technique

Structured light is a general concept and there are a number of ways of exploiting it to obtain 3-D information [25]. As such, a brief description of the structured light is "the process of illuminating an object with a specific light pattern and observing the lateral position of the image with a camera from a known angle" as shown in Figure 1.2. If a line of light is projected and viewed from one side, the distortions in the projected line can be translated into height variations along the line [26]. This technique is a simplified form of the stereophotogrammetry method and has been used by different groups for medical [27 ,28] or industrial applications [29]. The height extraction and accuracy of this technique will be elaborated on later.

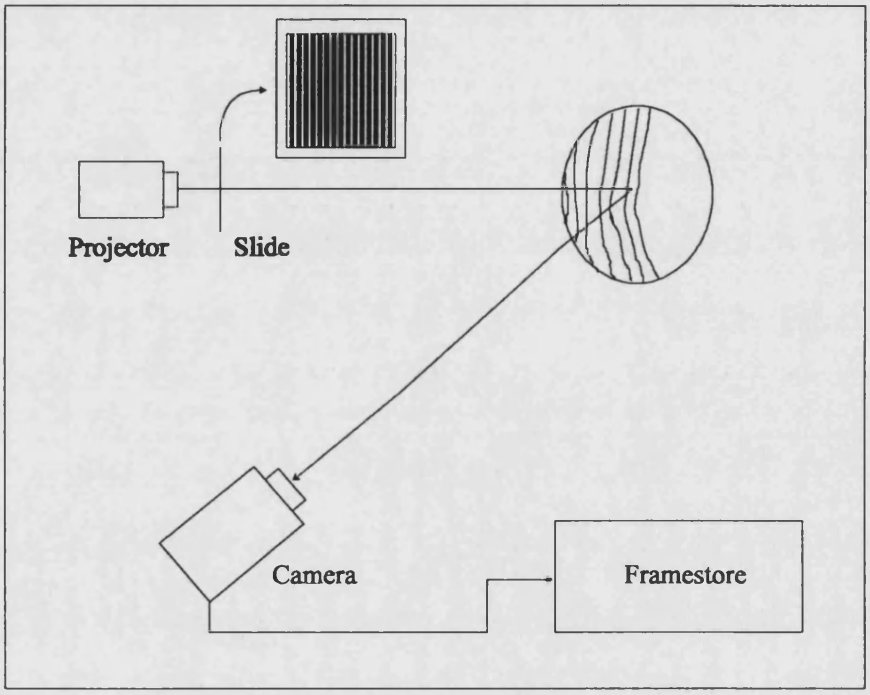


Figure 1.2: A static structured light system set-up

Two broad class of the structured light systems have been defined as parallel and serial techniques. Parallel techniques are currently the most applicable. These rely on modulating the entire visible surface of an object with a structured light pattern as explained previously, and then recovering range data for the entire surface using a single image as shown in Figure 1.3. Serial structured light techniques consist of devices which build up range maps by scanning, either using a laser beam or a time varying shutter. The laser time-of-flight range finder [30] is well known example of the serial structured light method.

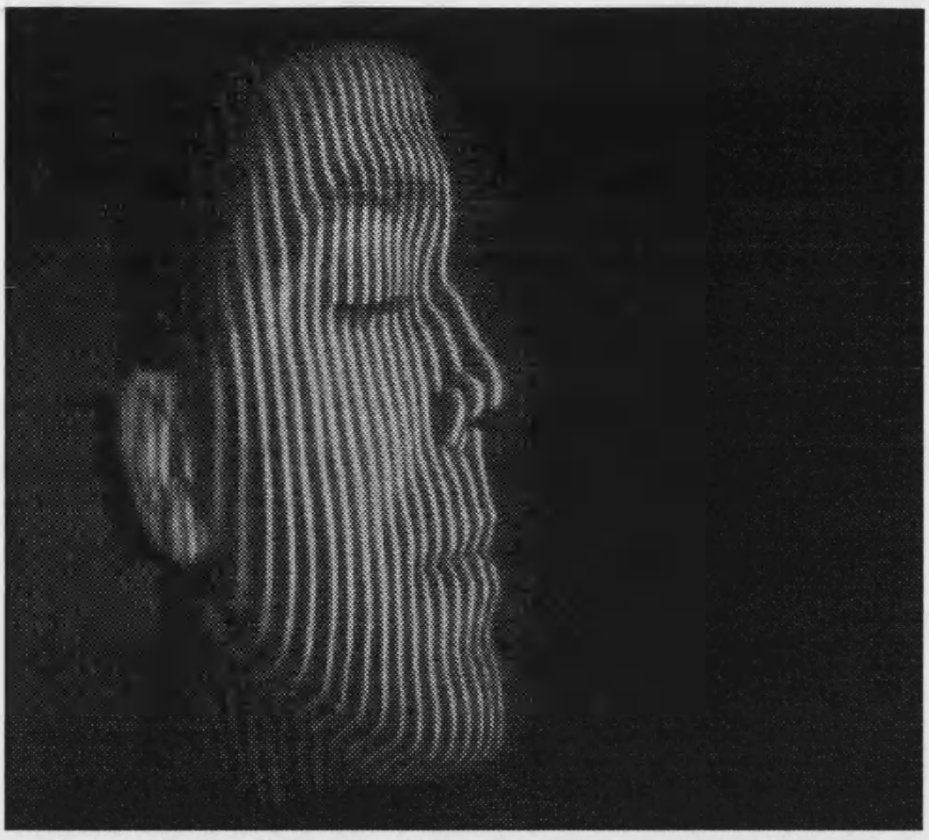


Figure 1.3: Structured light image by projecting a parallel vertical line pattern on the face

1.3.3 Raster Sterophotogrammetry

Raster stereophotogrammetry is a complementary technique to stereophotogrammetry, as shown in Figure 1.4. Here, one of the cameras has been replaced by a projector which projects a line or dot pattern onto the subject. The projection angle is controlled by the host microcomputer and the camera now views the pattern in a similar way as in the stereophotogrammetry method. However, because we know the projection angle and what the projection pattern looks like, the stereo correspondence problem is greatly simplified [31].

This technique has been deployed successfully by *Keefe et al* (1986) [32]. The drawbacks of such systems, however, are that mechanical moving parts are involved and therefore acquisition time and accuracy can be considerable. Also, because of the discrete arrays of images, results do not become apparent until the computer image processing is complete and therefore the clinicians does not get any feedback during screening.

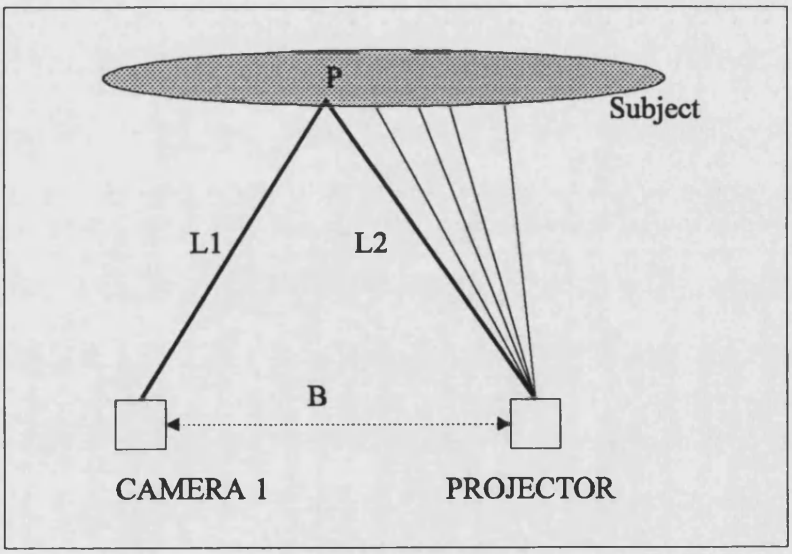


Figure 1.4: Plan of the raster stereophotogrammetry set-up

1.3.4 Integrated Shape Imaging System - ISIS

The Integrated Shape Imaging System (ISIS) or line stripe scanning system is a form of raster stereophotogrammetry [33]. The system comprises six parts; a console, the scanner (a covered 35 mm projector), a scanner platform, patient positioning frame, calibration board, and a camera as shown in Figure 1.5.

First of all the patient is positioned according to the calibrated frame and then the scanner projects a horizontal beam of light across the patient's body. The camera is mounted below the projector and positioned so as to be able to capture all the area scanned by the scanner. As the scanning line moves down vertically, frames from the video camera are digitised and stored. The number of frames captured depends on the

required accuracy with which the surface is to be mapped. This system, compared with other scanning techniques is very slow and was applied for only limited medical applications [34].

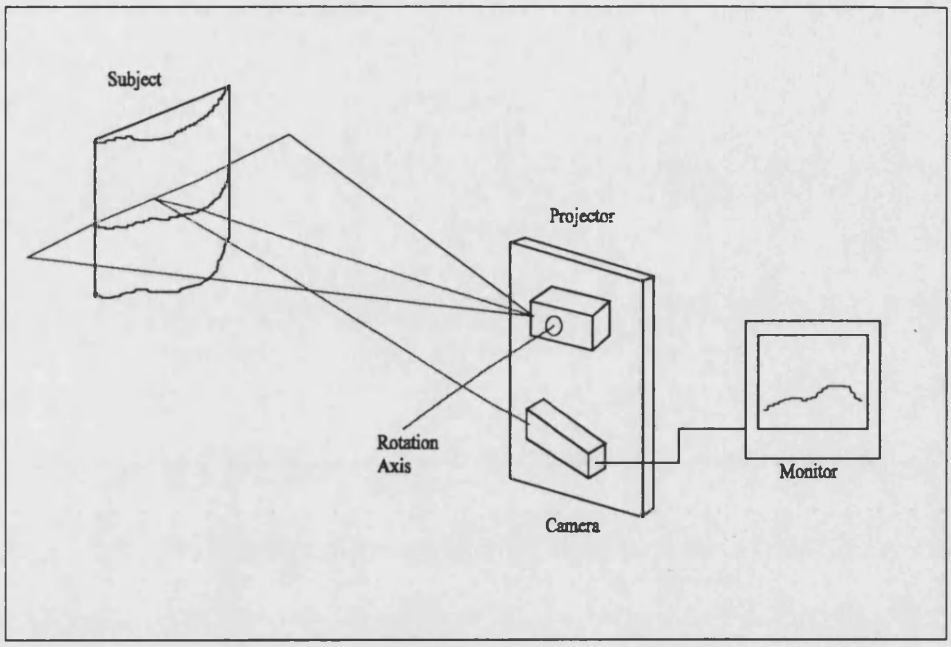


Figure 1.5: Integrated Shape Imaging System layout

1.3.5 Moiré Techniques

Moiré photography shows height information in the form of interference fringes (moiré fringes). These fringes are created and observed by one of two methods: projection moiré and shadow moiré. The first, projection moiré, is achieved by projecting a grid on the subject and viewing the resulting shadow cast through a grid of equal pitch as shown in Figure 1.6 (a). The second, shadow moiré, the subject stands behind a full size grid and is observed through the grid as shown in Figure 1.6 (b).

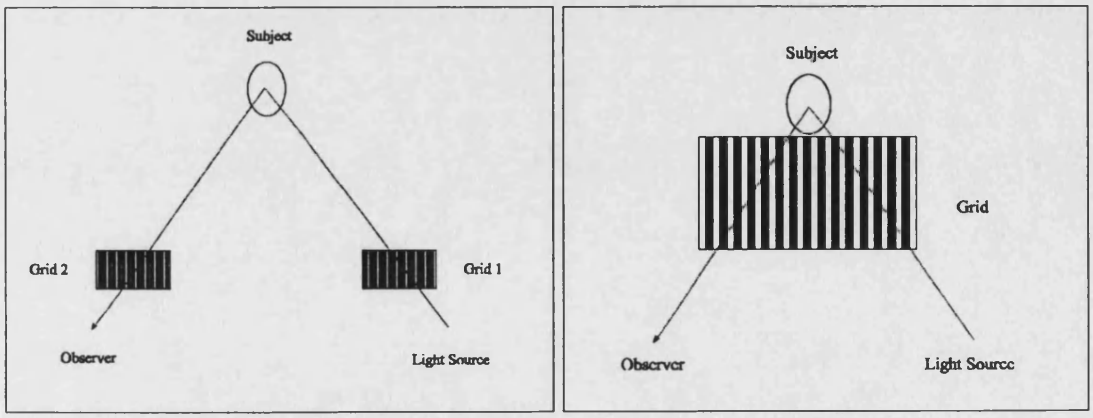


Figure 1.6: (a) Projection moiré, and (b) shadow moiré set-up

In both cases, moiré fringes are formed by interaction between a grid and the shadow of the grid, (or a similar grid). The fringes follow paths of constant height, but the fringe interval between neighbouring fringes is not constant and this makes the extraction of heights difficult. The fringe interval increases with the distance of the subject from the grid and is proportional to the pitch of the grid. This method was used by *Inokuchi et al* [35] for evaluation of facial palsy and assessing the recovery process but not for three dimensional reconstruction of the face. The moiré fringe contouring technique is limited by regions of shadow present on the object to be measured under the high levels of illumination required by the system. Therefore, when measuring each head from the front, careful positioning was required to eliminate shadows from the facial area. Figure 1.7 shows the moiré fringes on a model face captured by the implemented system. This was achieved by positioning the heads on their sides rather than upright as would normally be done behind a grid (shadow moiré).

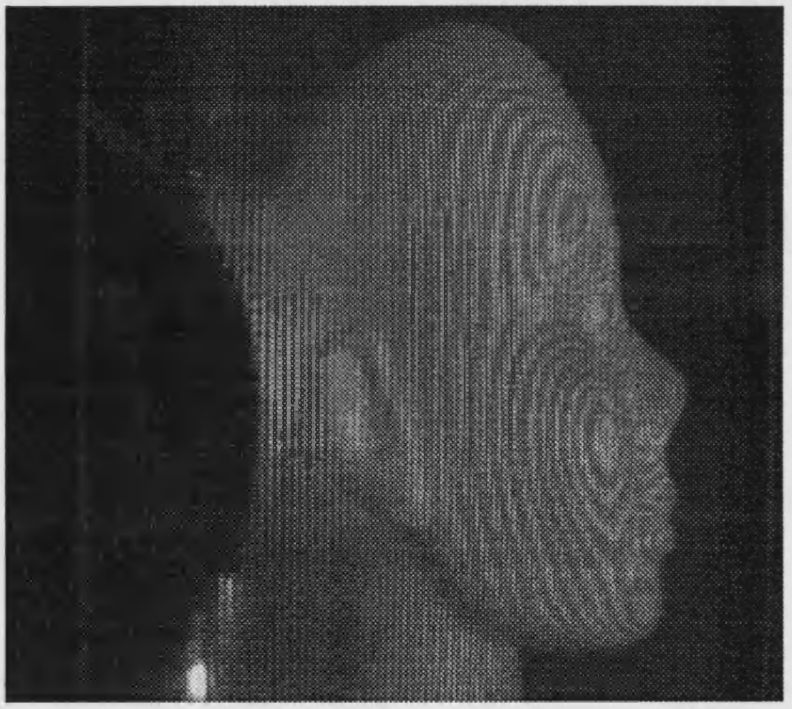


Figure 1.7: Moiré fringes on the model face

1.3.6 Phase Measuring Profilometry

Phase Measuring Profilometry [36], uses both raster stereophotogrammetry and the moiré technique. A sinusoidal grating structure is projected onto the subject and this image is captured using a CCD camera as shown in Figure 1.8. The grating is then phase shifted and another frame taken. Phase measuring algorithms [37] can compute the surface elevation very accurately. High-precision interpolation up to 1/1000 of a fringe period, yielding measuring precision ranging from 0.5 mm for the human trunk, to 0.1 mm for the face and 0.01 mm for the human teeth.

This method differs from raster stereophotogrammetry because a whole grating is projected in one go and then moved. The connection with moiré technique comes from the use of a sinusoidal grating, moiré type patterns can be very easily generated by multiplying the grabbed frames with a reference grid to give the qualitative data.
The major disadvantage of this technique is that the grid must be rotated with a very accurate mechanical system.

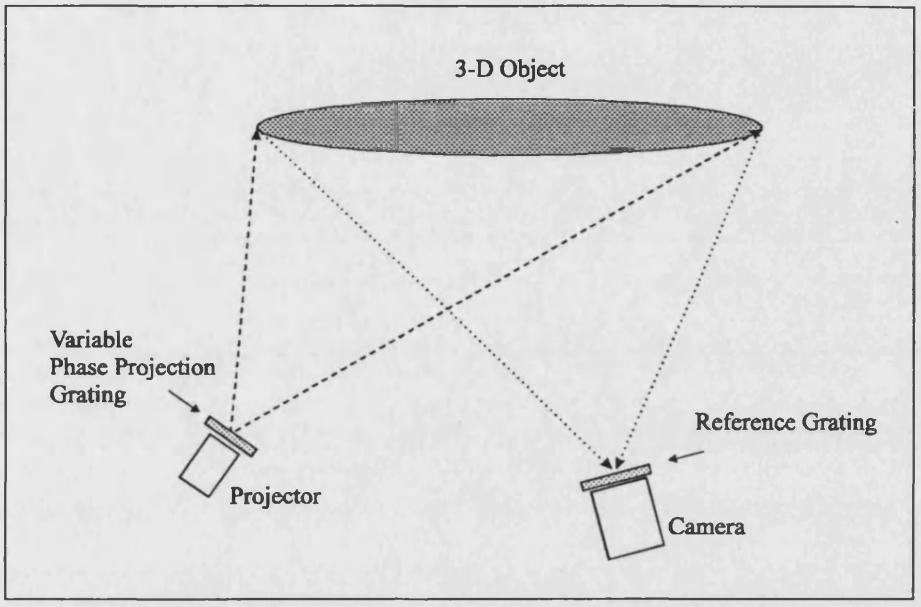


Figure 1.8:: Phase measuring profilometry set-up

1.3.7 Laser Scanning Method

In this technique a laser beam scans the subject's face. The reflected beam from the subject is detected by a CCD camera, the laser scanning movement and the camera detection are controlled by computer and so digital data can be easily obtained. The controller circuit moves the laser spot to a desired point and the controlling microcomputer uses simple trigonometry to establish where in the CCD's field of view the spot will fall. There will be a difference between the established and measured position which will be used to calculate the z co-ordinate.

Two types of laser scanning methods have been presented for capturing facial information by rotating the patient or moving the optical system as shown in Figure 1.9 and Figure 1.10. *Linney et al.* [20] implemented a laser system to compare images before and after facial surgery. In their system a laser beam is fanned into a vertical line using a cylindrical lens. The line is projected onto the patient's face and is then viewed obliquely by a video camera. The patient is rotated under computer control and the distortion of the laser line as it illuminates the face is recorded at every 2.8 degrees of rotation. The data is stored in computer memory and the approximately 20 000 co-ordinates on the facial surface are derived and the facial image with a precision of 0.5 mm reconstructed.

Young et al. [38] presented another type of the scanning system that moved the laser beam and the camera. The contour capture gantry consists of a trolley with two arms on it; one holds the camera and the other holds the laser projector. The motion of the trolley is controlled by a stepping motor with an accuracy of 1.86 mm. The projector projects a strip of light on the face, and CCD camera then captures the facial contours formed by the light.

Laser systems represent the ultimate in surface imaging accuracy but have the disadvantages of slow speed if the accuracy is to be realised, and require high precision mechanics.

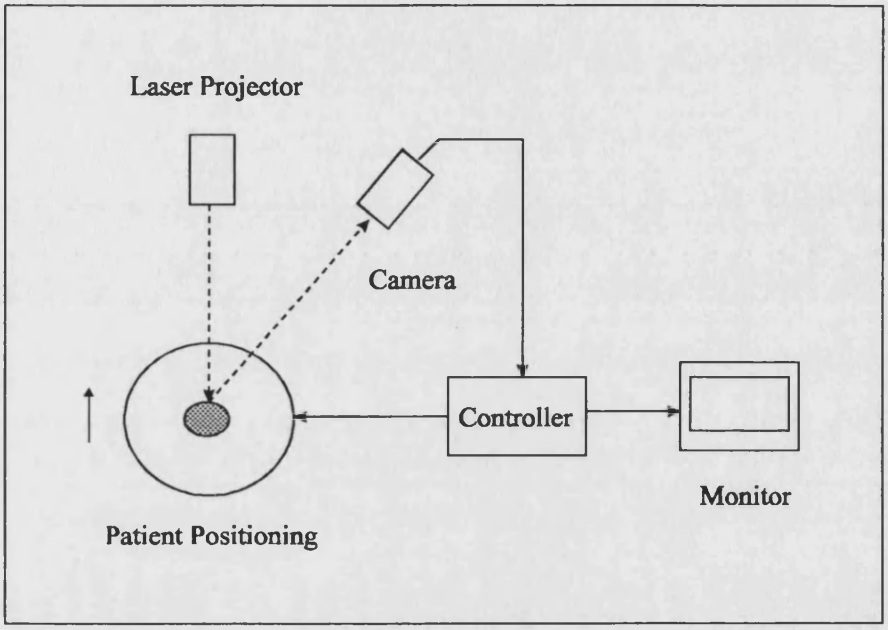


Figure 1.9: Laser scanning set-up by rotating the patient

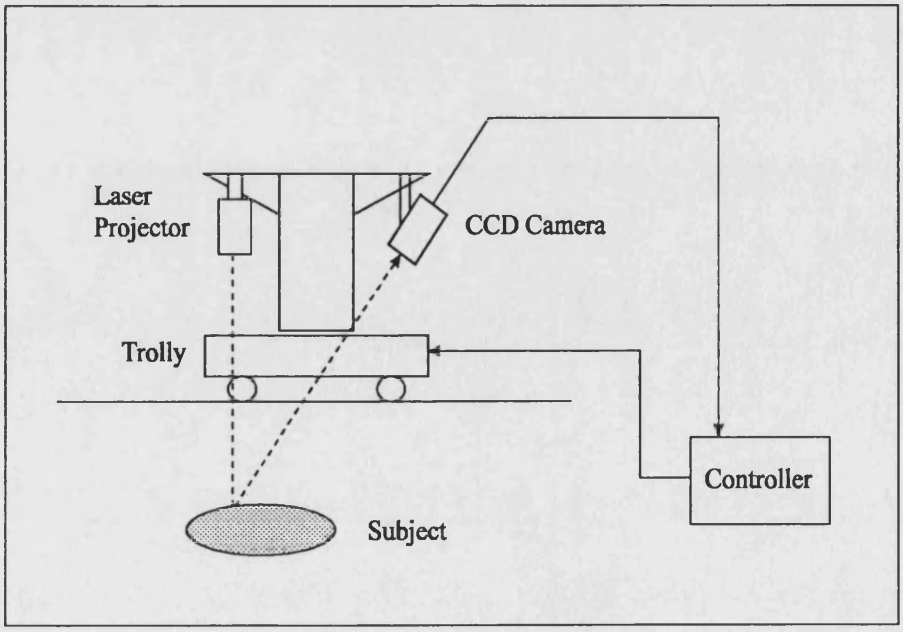


Figure 1.10: Laser scanning set-up by changing the position of the optic system

1.4 Discussion of techniques

Before discussing the merits of each system, it is necessary to set out the characteristics of a desirable system and then use that as a reference against which the above techniques will be judged.

The optimum system should be portable, easy to set up and fast. Accuracy about one millimetre (or less) is appropriate to detect the required information from the face. One of the aims of this work is to automate the processing of any acquired data and consequently a system that will need minimum operator training is required.

Table 1.1 shows a useful comparison of the discussed techniques.

System	Accuracy	Hardware cost	Speed	Portability	Applied Body area	Existing system applications
Raster	> 1 mm	Expensive	Minute	Fair	-	Robot Vision
Structured Light	> 1 mm	Inexpensive	Second	Good	All	Robot Vision
Moiré	> 3 mm	Inexpensive	Second	Fair	Back & Chest	Scoliosis Treatment
ISIS	> 3 mm	Expensive	Minute	Good	Back	Clinical Monitoring
Phase Measuring	< 1 mm	Expensive	Minute	Good	All	Human Body
Laser Scan	< 1 mm	Very Expensive	Minute	Fair	All	Human Body

Table 1.1: Comparison of the surface imaging techniques

The stereophotogrammetry method has the problem of calibrating the reference points. A suitable pattern may be projected onto the subject and the correspondence of each point on the both camera images and the projector can be found automatically. The considerable processing time makes this technique unacceptable for our application.

Raster stereophotogrammetry methods have the necessary accuracy, but it must be remembered that a live human body is the subject and any movement produces inaccuracy in the three dimensional reconstruction. Getting the system up to speed with useful accuracy requires precision mechanics to scan the structured light and appreciable power to process the data. Additionally, identifying the same point on both images can be difficult around the edges and in areas of low contrast. This means that pure raster stereophotogrammetric systems with these specifications are not appropriate.

The ISIS system has the same problems as the raster stereophotogrammetry method especially when the number of scanned lines is increased for greater accuracy. This system can be considered to be the optimum of raster systems.

Laser methods are the most accurate (less than 0.5 mm) of all techniques due to the ability of focusing a small powerful beam on the subject's face, but they have the same speed problems as raster stereophotogrammetry. Alignment of the laser and the camera is also very critical. However, a major problem with a laser system in facial use, apart

from the cost, is the health problem. When a laser beam is powerful enough to scan a face it can seriously damage the retina if accidentally pointed at eye. Using low power laser beams in order to protect the eye causes low accuracy of the facial image.

After laser scanning, the most accurate method is phase measuring profilometry. This has a measurement precision ranging from 0.5 mm for the human trunk to 0.1 mm for the face and therefore is suitable for capturing an image from a small area on the face. The major disadvantage of it is that the grid must be moved, for which high precision equipment is required.

The moiré fringe method is considered a fairly inexpensive technique but has serious problems when applied to the human face. Moiré contours from a single picture can not judge whether a concentric fringe represents a hill or a valley. Contours of moiré fringes on a living body have poor contrast due to the blurring of the shadow of the grating cast on the body. This method requires manual intervention for extracting height information, and also the accuracy of this technique is the lowest among the techniques considered because of the grating pitch and calibration problem for grates and patient. The method is commonly use for back surfaces where the spatial resolution required is lower, in order of 3-5 mm.

Among the many range finding methods, parallel structured light has been used widely because of its simplicity, speed, economy and safety. However, structured light has some limitations. It has problems with shadow areas and, not all 3-D points can be viewed in both camera and projector. This is common in all triangulation-based systems. This method can be easily implemented with available equipment such as a conventional camera and projectors.

1.5 Previous works on structured light technique for medical applications

The idea of regular patterns being distorted by irregular surfaces is not a new one. Since the middle of the last century workers have attempted to use the camera for producing solid sculptures of human subjects. *Francois Willeme* [39], a French painter and sculptor, took out patents for a method of photosculpture which involved photographing the subject's profile from twenty four different camera positions. Prints were produced and twenty four profile templates assembled to give a very crude reconstruction which formed the basis of a clay sculpture.

A significant improvement was made by *Poetschke* [39] when he proposed that a photograph should be taken of a slit of light falling on to the head. This lead *Givaudan* [39] to invent what he called "photostereotomy" where the subject was divided optically into a number of parallel slices by a slit of light. *Hertzberg* and *Saul* [40] then developed this technique using a "countourometer" for the US Air Force.

The original method of contour photography for medical use was invented by *Sassoni* [7] in 1957, he used a light sectioning technique for his "physioprint" which he hoped could be used in forensic applications for identity photographs. In Australia, *Roche et al.* [41] in 1962 designed a "cyrtographometer" which they used to study growth in

the female breast. *Cobb* [42] (1971) and *Lovesey* [43] (1972) produced equipment for calculating facial surface area and volume to help in the design of an air-crew oxygen mask. *Williams* [39] developed a double sided system for use in medical photography. This technique was evaluated by *Leivesley et al.* [44] for measuring facial deformities. *Dunn et al.* [28] in 1991 also applied this technique to the quantification of the area and volume of the surface burns. A new structured light method was introduced by *Ozturk et al.* [45] in 1996 to evaluate physical characteristics, such as perimeter, area, depth and volume of wounds.

Although the structured light technique has been used in several industrial applications for 3-D machine vision [46 , 47], its medical applications have been more limited because of the inaccuracies caused by skin colour and movement. Following the pioneering papers, doctors, engineers and scientists investigated the technique with the goal of achieving a fully automated system for 3-D imaging of the human body, but all of the designed systems required a significant amount of human intervention to extract height information.

Chapter 2

Structured Light System

The structured light system will be examined by explaining the designed pattern and mathematics for calculating the 3-D range information using the triangulation method, then the optical and hardware set-up will be explained in details.

2.1 Introduction to the structured light

The structured light optical methods are suitable for the measurement of the surface shape of objects when physical contact is undesirable. Usually these techniques rely on the projection of a pattern of light and dark stripes or grids onto the object in question, and the recording of one or more images of it. These techniques are often referred to as 'active illumination' or 'structured lighting' method. The projection of patterns onto the object is important for two reasons. First, the objects to be measured, certainly in the medical field, are often rather featureless. The projected patterns enhance features of the images and allow measurements to be made. Secondly, the pattern reduces the complexity of the stereoscopic matching problem. In any stereoscopic method the correspondence problem must be solved. In techniques where a stereo pair of images is recorded, the correspondence problem consists of discovering pairs of points, one in each image, resulting from the same object feature. In techniques where one of the cameras is replaced by a projector, as in the structured light method, the correspondence problem is that of knowing which part of the projected pattern gave rise to a particular image feature [46].

Parallel structured light techniques can broadly split into two main categories, those which are stripe-based (1-D pattern) [48], and those which contain coding in a second dimension (2-D pattern) [49], such as spots or a grid. Stripe-based techniques have the advantages of (a) being able to provide a high sampling rate along the length of the stripes, which can be aligned with the surface to be measured in the most favourable manner and (b) having some benefit in robustness during line extraction by enforcing the local continuity. 2D patterns such as spot-based techniques suffer a low sampling rate in both dimensions, but are potentially more robust due to redundancy in the system [50].

Two categories are available for generating the structured light pattern; monochrome or colour pattern. The use of colours to code the light stripes solves the correspondence problem, but the complexity of the optic and acquisition system will be increased. The effects of skin texture and colour variations are also very important for the colour based system.

To improve the accuracy of the structured light system and especially to help solve the correspondence problem, a technique based on codifying the projected pattern has been proposed [51 , 52], so that each projected light point carries some information. Many patterns with a special coding may be applied for capturing the surface information. These type of patterns are limited to static scenes with motionless objects.

For our implemented system (BSLS) a novel 2-D structured light pattern was selected; the brick pattern as shown in Figure 2.1. The reason for choosing this pattern was: (1) to have only one projected pattern for instantly recording the scene in order to reduce the inaccuracy from patient movement, and (2) to increase the labelling and interpretation accuracy (this will be discussed later in chapter 5), in order to have more samples from the facial surface.

The structured light imaging system is shown in Figure 2.1, the system works on the basis that if a known pattern of light is projected onto an object then the object is viewed from an offset angle, the reflected light produced is a distorted version of the original pattern. This distortion is directly related to the surface of the subject's face and so it is possible to calculate the contours of the subject's surface by comparing the distorted image with the expected projection of the original pattern. The system is designed to capture two sides of a face separately by projecting a special slide for each one.

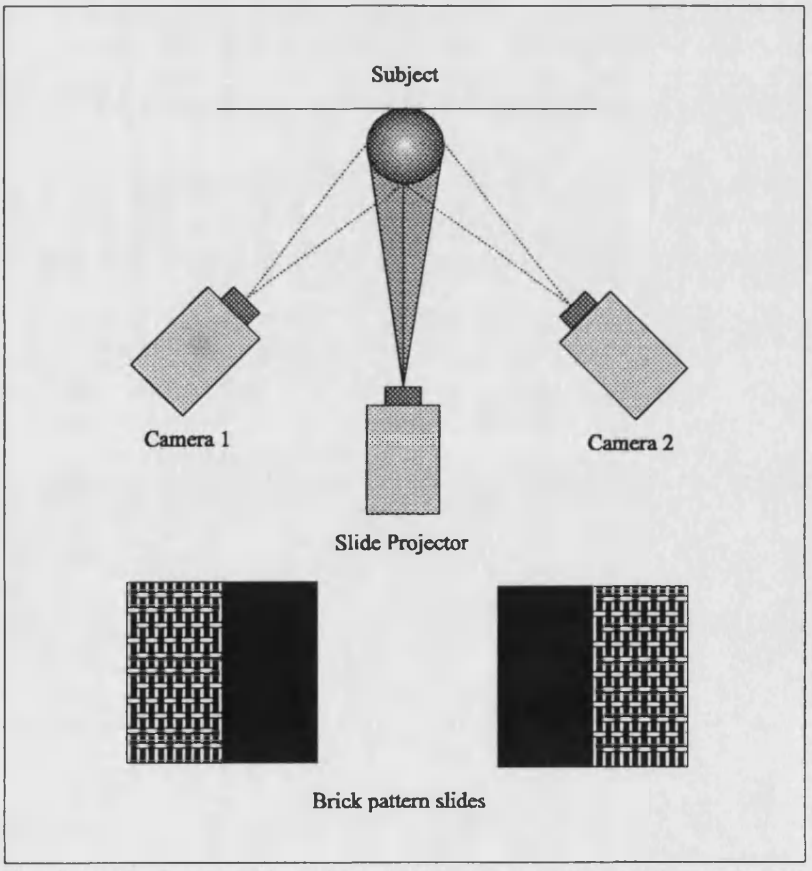


Figure 2.1: A representation of the structured light system, one pattern for each side of a face (the distance between each vertical line is d_g and the distance between each horizontal line is $2d_g$).

2.2 Principles of the structured light 3-D vision

The process of 3-D reconstruction consists of applying simple geometry to find the intersection of the camera rays corresponding to the detected features with the appropriate plane of the projected light, and is both trivial and error-free. The 3-D profile of the object under measurement is obtained with respect to a base plane, which acts as a plane with a defined y co-ordinate as shown in Figure 2.2. The z axis is considered on the projected line and the y axis is perpendicular to the plane of x-z. The height from the base plane for point (H) on the subject's surface can be determined by comparing the triangles. The plane $z=0$ is known as the 'base plane'. The triangulation formula depends on the optical geometry of the system, the optical axis of the acquisition unit, and the intersection of the optical axis with the projection line in point (H). Three parameters define the triangulation geometry. These are the distance (d_{cp}) between the camera and the projector, the distance (d_c) between the camera and the base plane and the correspondence shift ($H'S$) in the base plane. d_p is the projector distance from the base plane.

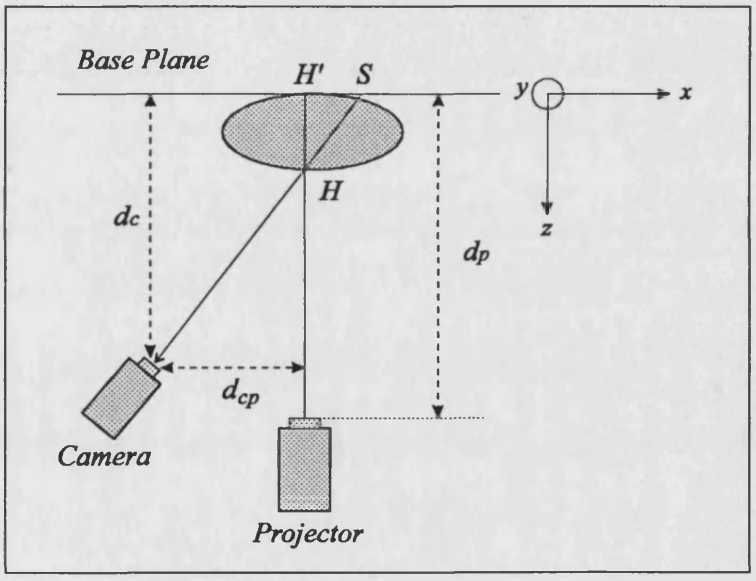


Figure 2.2: Simple geometry of the structured light optic system for the left camera

This geometry results in the following triangulation formula:

$$H'H = \frac{d_c \cdot H'S}{d_{cp} + H'S}$$ Equation 2.1

The above equation allows the evaluation of the height ($H'H$) with respect to the base plane. Moreover, the co-ordinates of point $H(x_H, y_H)$ are expressed with reference to the co-ordinates of the corresponding image point on the image plane. These co-ordinates define the Camera Recording Plane, or image plane. The first (x, y, z) co-ordinate is fixed to the patient positioning and the second (x', y', z') co-ordinate is associated with the subject image in the camera recording plane. The (x_g, y_g) co-ordinate is defined on the slide to show the projection point.

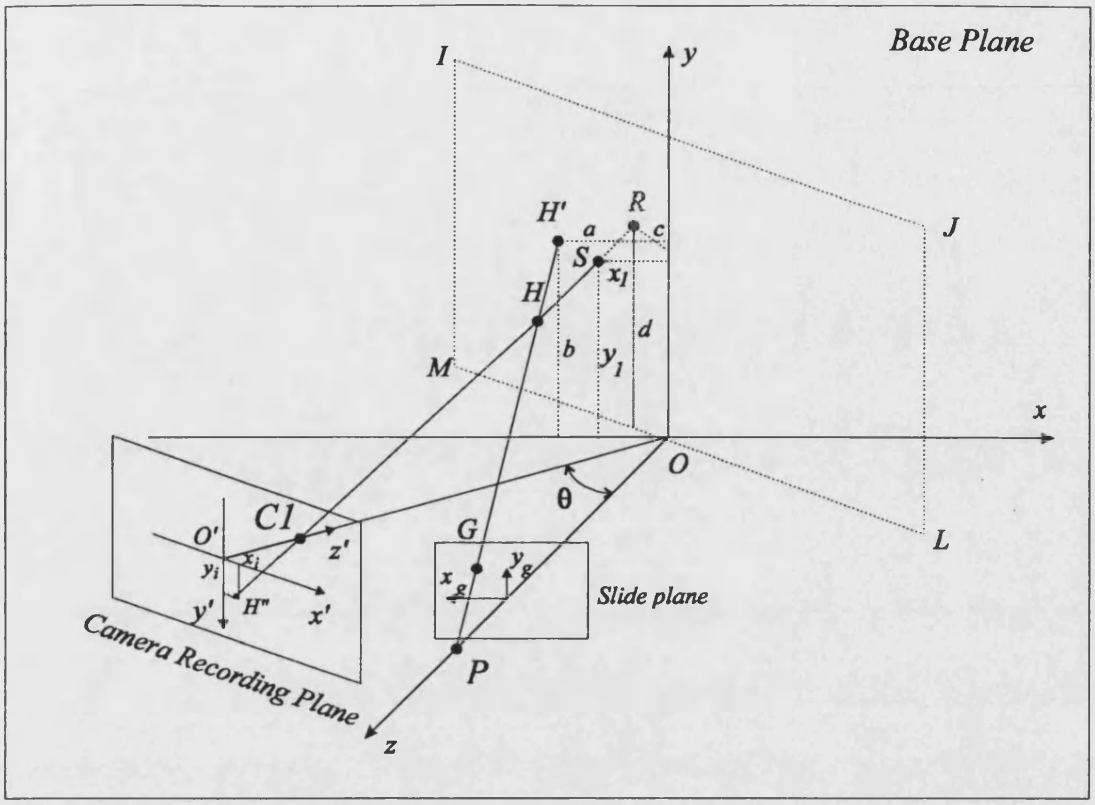


Figure 2.3: Structured light geometry for the left camera (C_1), H is a point on the subject's face illuminated by the projector (P)

The geometry related to the left hand camera of the system (C_1) is shown in Figure 2.3. H, with co-ordinates (x_H, y_H, z_H), represents a point on the object. The pattern ray from the projector (P) illuminates this point . C_1H represents the path of the reflected light ray from the object to the camera. S is the correspondence shift point on the base plane with the co-ordinate (x_1, y_1). The image point on the camera recording plane is $H``$ with the co-ordinate (x_i, y_i).

Lines IJ and ML lie in a plane perpendicular to the camera optic axes (OC_1) and therefore parallel with the camera recording plane. The co-ordinates of point R on this plane are related by translation and scaling factors to the pixel co-ordinates of the image point H in the camera ($H``$). These factors can be determined by including fiducial marks in the apparatus at known positions [53]. Point R, (c, d) on the $IJLM$ plane, actually lies at the point ($c.\cos\theta$, d, $c.\sin\theta$) in true (x, y, z) space after applying the rotation matrix, θ is the angle between OC_1 and the positive z axis. To find the correct relationship between the image point and point R, it is necessary to consider the triangles, the result is,

$$\begin{cases} c = \dfrac{d_c x_i}{f_c \cos\theta} \\ d = \dfrac{d_c y_i}{f_c \cos\theta} \end{cases}$$

Equation 2.2

where f_c is the camera focal length ($O`C_1$).

The equation 2.2 can be written by considering the scale and optic amplification factors,

$$\begin{cases} c = \beta_c s x_i / \cos\theta \\ d = \beta_c s y_i / \cos\theta \end{cases} \qquad \text{Equation 2.3}$$

where s is the scaling factor for the pixel co-ordinates in the camera with the relative co-ordinates in the computer screen and the parameter β_c is the camera amplification factor,

$$\beta_c = d_c / f_c \qquad \text{Equation 2.4}$$

Points C_1, H and R are colinear and the line C_1R intersects the plane $z=0$ at the point S whose co-ordinates are $(x_1, y_1, 0)$. Since C_1 has co-ordinates $(-d_{cp}, 0, d_c)$, it is clear that the equation of line C_1S is given by,

$$\left(\frac{x + d_{cp}}{x_1 + d_{cp}}\right) = \left(\frac{y}{y_1}\right) = \left(\frac{z - d_c}{-d_c}\right) \qquad \text{Equation 2.5}$$

Using the fact that point R also lies on this line and using the equation 2.3, the co-ordinate of the corresponding shift point will be obtained,

$$\begin{cases} x_1 = \left(\dfrac{-\beta_c s x_i (d_{cp}\tan\theta + d_c)}{\beta_c s x_i \tan\theta - d_c}\right) \\ y_1 = \left(\dfrac{-\beta_c s y_i d_c}{\cos\theta(\beta_c s x_i \tan\theta - d_c)}\right) \end{cases} \qquad \text{Equation 2.6}$$

Similar expressions can be developed to calculate the projector ray line equation. The points P, H and H' are also colinear. Since the projector ray has the co-ordinate $P(0,0,d_p)$ and the projector ray on the base plane can be defined with the co-ordinate (a, b), the equation of the line PH' is,

$$\left(\frac{x}{a}\right) = \left(\frac{y}{b}\right) = \left(\frac{z - d_p}{-d_p}\right) \qquad \text{Equation 2.7}$$

The co-ordinates of the point H satisfies the equations 2.5 and 2.7 and can be shown as,

$$\begin{cases} \left(\frac{x_H + d_{cp}}{x_1 + d_{cp}}\right) = \left(\frac{y_H}{y_1}\right) = \left(\frac{z_H - d_c}{-d_c}\right) \\ \left(\frac{x_H}{a}\right) = \left(\frac{y_H}{b}\right) = \left(\frac{z_H - d_p}{-d_p}\right) \end{cases}$$ Equation 2.8

Now the co-ordinates of the point H (x_H, y_H, z_H) on the face can be calculated by applying the parameters (a, b, x_1 and y_1). To find the parameter a or b, we can use one of the defined models for the projector. A simple pinhole model for the projector is shown in Figure 2.4. For modelling the projector rays, the points on the vertical parallel lines are considered. The line width on the slide is important for the total accuracy, small width means more surface resolution. The focal length of the projector effects the expected projection in a significant manner because it governs the width of the structured light lines on the plane of the face. The distance between two projected lines on the base plane (d_o) can be calculated,

$$d_o = d'_W + d'_B = (d_W + d_B) \cdot (1 + \frac{d_p}{f_p + \varphi_p})$$ Equation 2.9

where (f_p) is the projector focal length, (φ_p) is the projector optic path so that ($d'_p = f_p + \varphi_p$), (d_p) is the projector distance from the base plane, d_W is the slit width of the slide and d_B is the black distance between two gaps on the slide. The distance between two vertical lines on the slide (d_g) is,

$$d_g = d_W + d_B$$ Equation 2.10

For a point r when it lies on the k^{th} projected vertical line, its distance from the reference line will be,

$$d_r = kd_o = kd_g\beta_p$$ Equation 2.11

where β_p is the projector amplification factor,

$$\beta_p = (1 + \frac{d_p}{f_p + \varphi_p})$$ Equation 2.12

For point H' (a, b) on the projector ray, we can find a corresponding point on the slide plane (G) with the co-ordinate of (x_g, y_g) so that,

$$\begin{cases} a = \beta_p x_g \\ b = \beta_p y_g \end{cases}$$ Equation 2.13

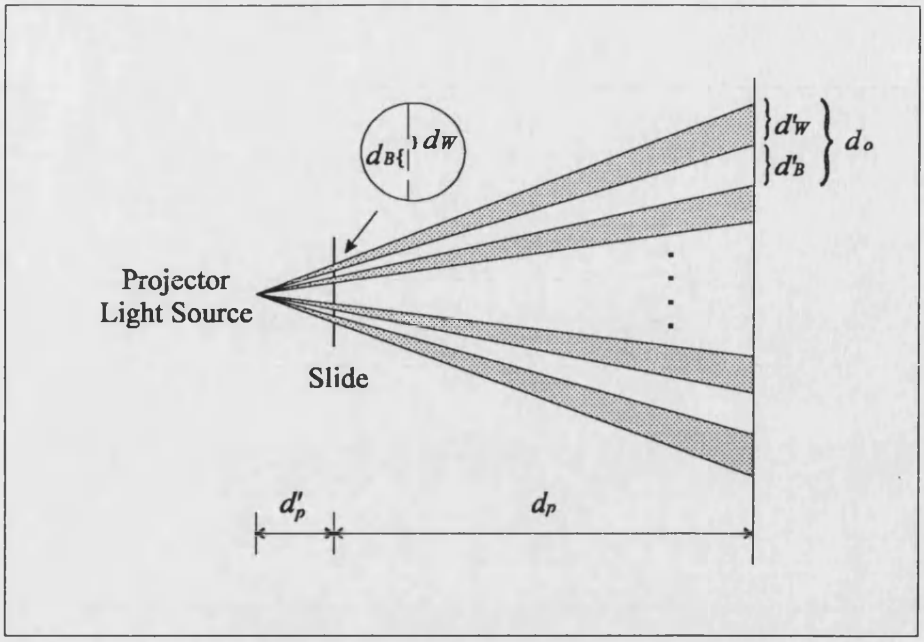


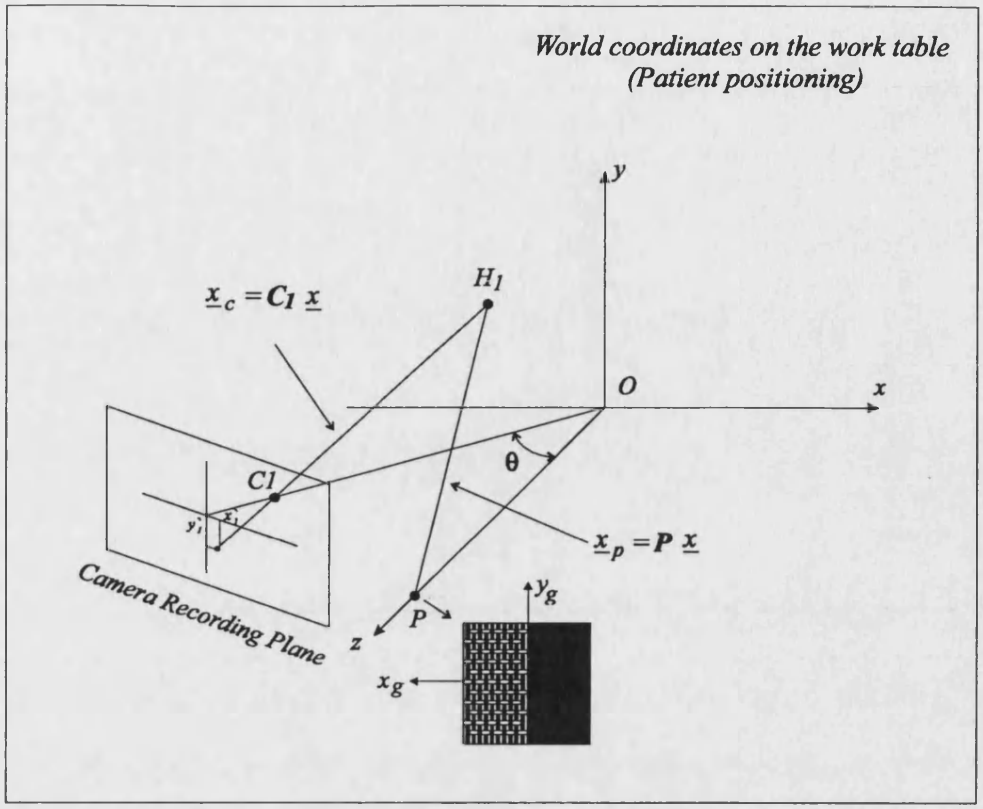
Figure 2.4: The projector pinhole model for calculating the width of lines

The equation 2.11 can be extended for other points on the projected lines along the x axis. This is achieved by relating each of the labelled lines on the image to one of the vertical lines on the grid. Each individual line represents a slice through the subject in the base plane. To identify the individual lines, a line number is assigned by considering its position from the reference line.

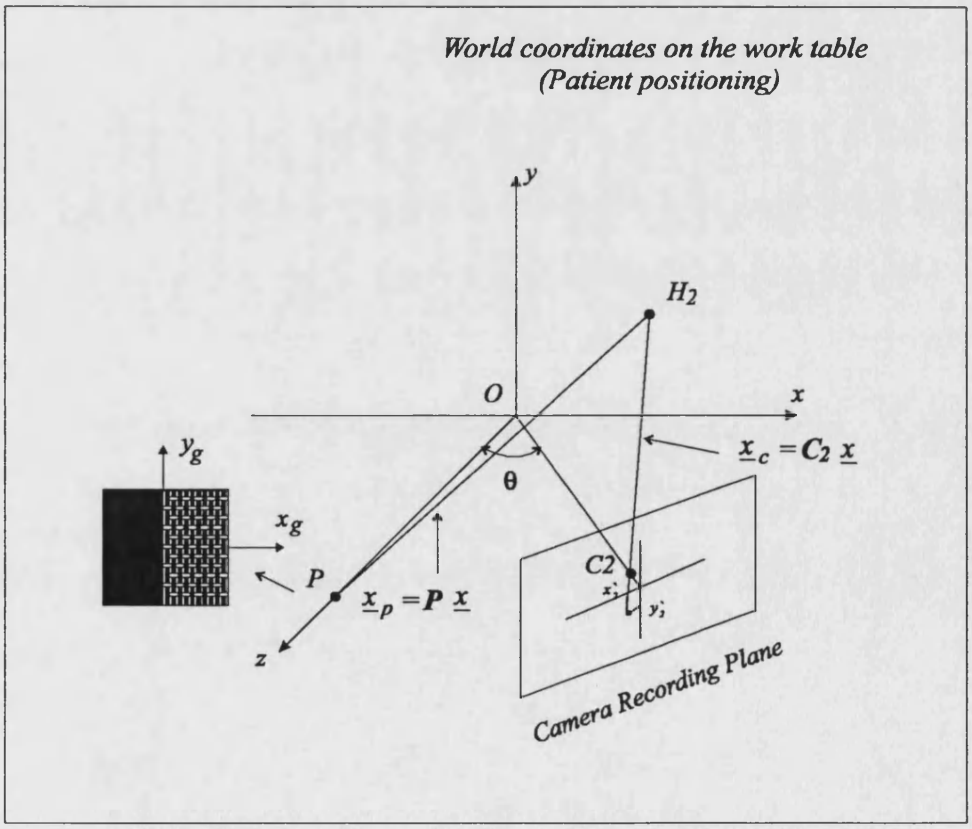
2.3 Homogeneous Co-ordinates and Calibration Matrices

Homogeneous co-ordinates are a mathematical representation with which one can express the perspective transformation as a linear transformation [53]. In homogeneous co-ordinates, a 3-D Cartesian vector (x, y, z) is represented by a four-component vector (wx, wy, wz, w) where w is an arbitrary constant. Two homogeneous vectors differing only by a scale factor represent the same physical point. The Cartesian co-ordinates of a point are recovered from the homogeneous co-ordinates by dividing each of the first three components by the last component.

The geometric co-ordinates of the system for the both cameras (left and right) and the projector are illustrated in Figure 2.5. The world co-ordinate system (x, y, z) is fixed on the patient positioning. The image co-ordinate system of the camera has origin at the centre of the lens. The image is digitised and displayed on a monitor with x axis from left to right and y axis from top to bottom (512×512). The projector co-ordinate system is similar to the camera co-ordinate system, except that the unit is grid line number and their junction points rather than pixel.



(a)



(b)

Figure 2.5: Homogeneous co-ordinates and the camera and projector vectors, (a) the left camera (C_1) and (b) right camera (C_2).

The camera image formation process which maps the point (x, y, z) in the object-centred co-ordinates onto the point (x_i , y_i) in the camera image plane (camera recording plane) can be represented by the following matrix equation in homogeneous co-ordinates,

$$\begin{bmatrix} w_c x_i \\ w_c y_i \\ w_c \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} & c_{13} & c_{14} \\ c_{21} & c_{22} & c_{23} & c_{24} \\ c_{31} & c_{32} & c_{33} & c_{34} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \qquad \text{Equation 2.14}$$

where c_{11}, c_{12}, ..., c_{34} are the elements of the camera calibration matrix (C).

This system of three linear equations in the homogeneous co-ordinates is equivalent to the following system of two equations involving the Cartesian co-ordinates,

$$\begin{bmatrix} c_{11} - x_i c_{31} & c_{12} - x_i c_{32} & c_{13} - x_i c_{33} & c_{14} - x_i c_{34} \\ c_{21} - y_i c_{31} & c_{22} - y_i c_{32} & c_{23} - y_i c_{33} & c_{24} - y_i c_{34} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \qquad \text{Equation 2.15}$$

The camera calibration matrix represents the functional relationship between points in the 3-D object-centred co-ordinate system and the pixel locations of the projected points in the camera image plane. The camera calibration matrix is a 3 by 4 matrix that represents the composition of three operations in the camera image formation process [54].

1. Rotation and translation between the camera and the object-centred co-ordinate axis.
2. Perspective transformation of the object point onto the camera image plane.
3. Scaling prior to digitisation of the camera image.

The goal of the camera calibration procedure is to solve for the unknown elements c_{11}, c_{12}, ..., c_{34} of the camera calibration matrix.

The projector calibration matrix (**P**) describes how the grid slide is projected into space. The elements of the projector calibration matrix depend on the position and orientation of the projector relative to the object-centred co-ordinates. The same procedure as explained for the camera calibration may be defined for the projector.

$$\begin{bmatrix} w_p x_g \\ w_p y_g \\ w_p \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \qquad \text{Equation 2.16}$$

where p_{11}, p_{12}, ..., p_{34} are the elements of the projector calibration matrix.

The projector matrix can be found by defining a suitable model for the projector and then calculating the light plane for the defined model. For a simple pinhole model and by considering the equation 2.7, the calibration matrix is obtained as,

$$\underline{\mathbf{P}}=\begin{bmatrix} -d_p & 0 & 0 & 0 \\ 0 & -d_p & 0 & 0 \\ 0 & 0 & \beta_p & -\beta_p d_p \end{bmatrix} \qquad \text{Equation 2.17}$$

As defined previously, triangulation is the method by which the 3-D location of points on the face can be determined as the intersection of two lines: one line that passes through the camera lens centre and the image of point on the face in the camera recording plane, and a second line that passes through the projector lens centre and the matching point on the slide. In practice these two lines usually come close to intersection but do not actually meet. This is due to quantisation of the digital image and numerical inaccuracies in determining the camera and projector calibration matrices. The usual approach to solve this problem is to find the least-squares intersection of the two lines, which can be defined as the point in space that minimises the sum of the squares of distances to the two lines. This point can be found by the following procedure.

Let $c_{11}, c_{12}, \ldots, c_{34}$ be the elements of the camera calibration matrix and $p_{11}, p_{12}, \ldots, p_{34}$ be the elements of the projector calibration matrix. Assume that the image plane co-ordinates (x_i, y_i) of a detected line point and the line label point (x_g, y_g) of the matching line on the slide are known. Then the least-squares intersection of the two lines (x, y, z) determining the object point is given by the least-square solution [55] of the following set of four linear equations in the three unknowns:

$$\begin{bmatrix} c_{11}-x_i c_{31} & c_{12}-x_i c_{32} & c_{13}-x_i c_{33} \\ c_{21}-y_i c_{31} & c_{22}-y_i c_{32} & c_{23}-y_i c_{33} \\ p_{11}-x_g p_{31} & p_{12}-x_g p_{32} & p_{13}-x_g p_{33} \\ p_{21}-y_g p_{31} & p_{22}-y_g p_{32} & p_{23}-y_g p_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} x_i c_{34}-c_{14} \\ y_i c_{34}-c_{24} \\ x_g p_{34}-p_{14} \\ y_g p_{34}-p_{24} \end{bmatrix} \qquad \text{Equation 2.18}$$

This procedure gives a set of points (x, y, z) on the face that can be used to reconstruct the surface.

The explicit equations have been developed for calculating the illuminated facial points. Implementation of these equations requires knowledge of the focal length, distances and offset angle. Although these parameters could be measured directly, determining one or more of the parameters using the camera itself as a measuring device often is more convenient (especially when the camera moves frequently). This requires a set of image points whose world co-ordinates are known, and the computational procedure used to obtain the camera parameters using these known points is often referred to as camera calibration. In general if we know the co-ordinates of six or more noncoplanar object points (x, y, z) and the co-ordinates of their corresponding image points (u_i, v_i) on the camera recording plane, the unknown camera calibration matrix elements can be

found by the least-squares solution of the overdetermined linear system of equations [56] as shown below, (c_{34} is considered 1 and the calculation can be done for the remaining 11 elements of the calibration matrix).

$$\begin{bmatrix} x_1 & y_1 & z_1 & 1 & 0 & 0 & 0 & 0 & -u_1x_1 & -u_1y_1 & -u_1z_1 \\ x_2 & y_2 & z_2 & 1 & 0 & 0 & 0 & 0 & -u_2x_2 & -u_2y_2 & -u_2z_2 \\ x_3 & y_3 & z_3 & 1 & 0 & 0 & 0 & 0 & -u_3x_3 & -u_3y_3 & -u_3z_3 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n & y_n & z_n & 1 & 0 & 0 & 0 & 0 & -u_nx_n & -u_ny_n & -u_nz_n \\ 0 & 0 & 0 & 0 & x_1 & y_1 & z_1 & 1 & -v_1x_1 & -v_1y_1 & -v_1z_1 \\ 0 & 0 & 0 & 0 & x_2 & y_2 & z_2 & 1 & -v_2x_2 & -v_2y_2 & -v_2z_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & x_n & y_n & z_n & 1 & -v_nx_n & -v_ny_n & -v_nz_n \end{bmatrix} \begin{bmatrix} c_{11} \\ c_{12} \\ c_{13} \\ c_{14} \\ c_{21} \\ c_{22} \\ c_{23} \\ c_{24} \\ c_{31} \\ c_{32} \\ c_{33} \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \\ v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix} \quad \text{Equation 2.19}$$

The calibration of the camera and projector by considering the lens distortion and defining a suitable model for the projector are explained in chapter 6.

2.4 Biostereometric Structured Light System Set-up

The biostereometric structured light system, as shown in Figure 2.6, used two CCD cameras to capture images from each side of a face and a slide projector to project a chosen grid. Two CCD cameras with 685(H) × 585(V) pixels resolution and precision zoom lenses were used as this type of camera gives a good low-light response. This type of camera has automatic gain control (AGC) and always it tries to compensate for low light conditions and because of this, the captured image looks artificially bright, and noisy. Therefore the hardware was designed to allow the user to manually set the CCD gain.

A 35 mm slide projector with 70-120 mm zoom lens projects a grid of parallel white lines on each side of the patient's face. The focusing and levelling of the projector are important to capture a reasonable image, and it should be done by projecting the reference slide on the face.

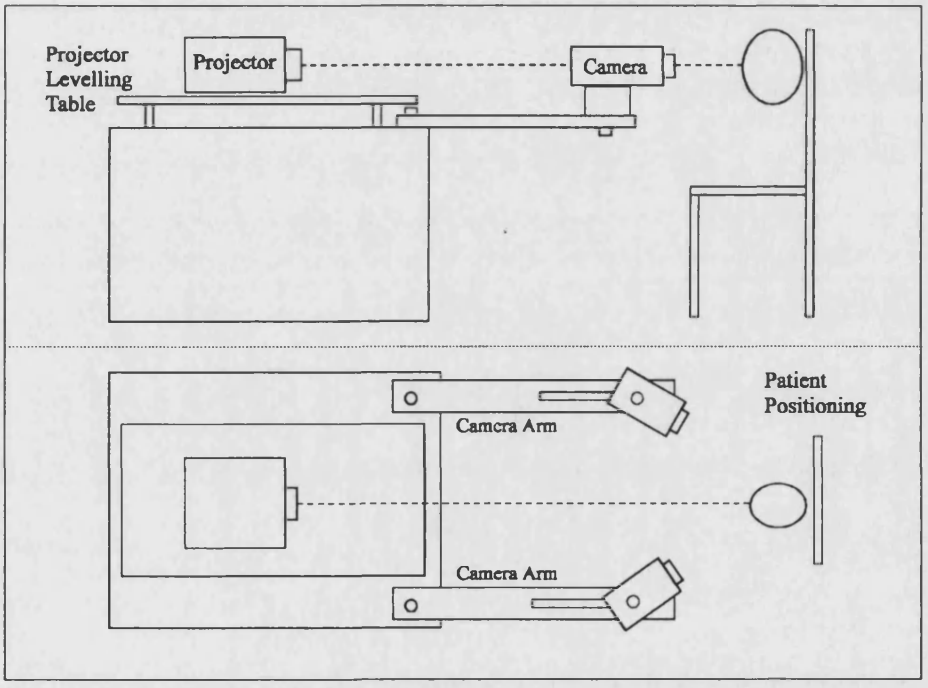


Figure 2.6: The layout of the designed system. The distance of the projector and cameras from the face, the separation angle, and levelling the projector should be done before capturing the facial image.

One of the important parts for this system is the grating through which the light is projected to produce the desired pattern. The pattern of structured light was originally created by using a computer art package, printing out the result and then photographing it with a film as a slide. But, in practice, it was found that regular photographic slides were not suitable because the dark parts of the slide were insufficiently opaque and caused a poor contrast level in the image, and also it was deformed by the projector lighting system. This problem was solved by using a lithography-on-glass technique instead of slide photography. The new slide generates sharp stripes on the subject's face and the result is high quality images.

2.5 Hardware requirements

As well as the optic set-up, a framestore is required to get the images into a computer for image processing. The framestore must meet the following requirements:

- It must acquire the image in one single frame period. This is because any slight movement could compromise the accuracy of the results.
- Horizontal and vertical resolution must be high for recording the height detail because the framestore must resolve to sufficient detail.
- A succession of grabbed frames should be possible. Clinical use requires several images to be grabbed and previewed so that a suitable image with regard to equipment set-up and patient positioning is achieved.
- The framestore should have industry standard connection for rapid processing on computers.

With these requirements, a high resolution, standard interface frame-store hardware card was designed and built. The frame-store card is plugged into the PC directly and can grab a frame with 512 × 512 pixel resolution and 256 grey levels in real time. There are many frame-stores commercially available but the main reasons for a custom design were;

- Multi-camera-input facility
- Manual gain control for compensating the camera intensity
- Automatic optic system adjusting and levelling
- Preview facility

The clinical use requires a suitable image with regard to equipment set-up and patient positioning. The patient positioning problem is particularly important as it must be remembered that people tire quickly when asked to sit motionless for even the shortest times. To this end, the framestore has a hardware preview facility whereby an image can be viewed instantly before grabbing, and the intensity of the camera and patient positioning may be adjusted for optimum results. Also the camera and projector distances, angle and other necessary calibration process before grabbing the image should be done by using the preview facility.

2.6 Calibration of the optical set-up

The first step for capturing the image is focusing and levelling the projector by using a reference slide. This slide is a cross pattern with a vertical and horizontal lines in the centre of the slide. The centre of the cross pattern is projected onto the highlighted point at the middle of the face and the vertical line at the middle of the forehead, nose and chin as shown in Figure 2.7. The patient positioning must also be calibrated at this point. Any tilt or unfocusing problem on the projected line will cause problems in matching of the reconstructed images.

The accuracy of projecting and adjusting the reference pattern onto the profile is related to the accuracy of the highlighted points on the face. Three reference points on the face have been selected for projecting the vertical line, middle of mouth, nose and between two eyebrows. The method has been used for high lighting these points is the same method as using by opticians for measuring the eyes parameters, measuring the distance between two points and then finding the middle point.

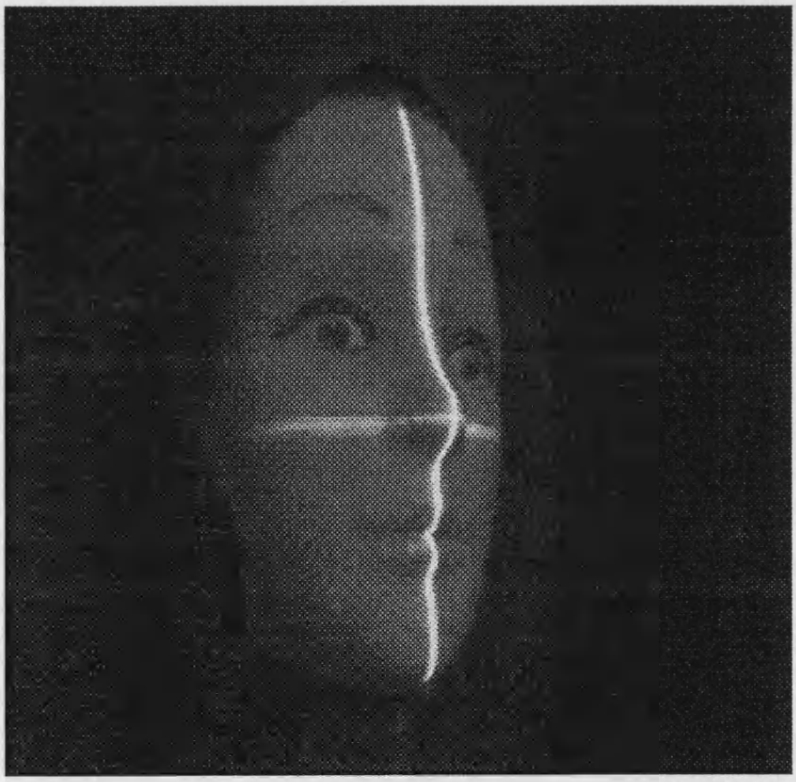
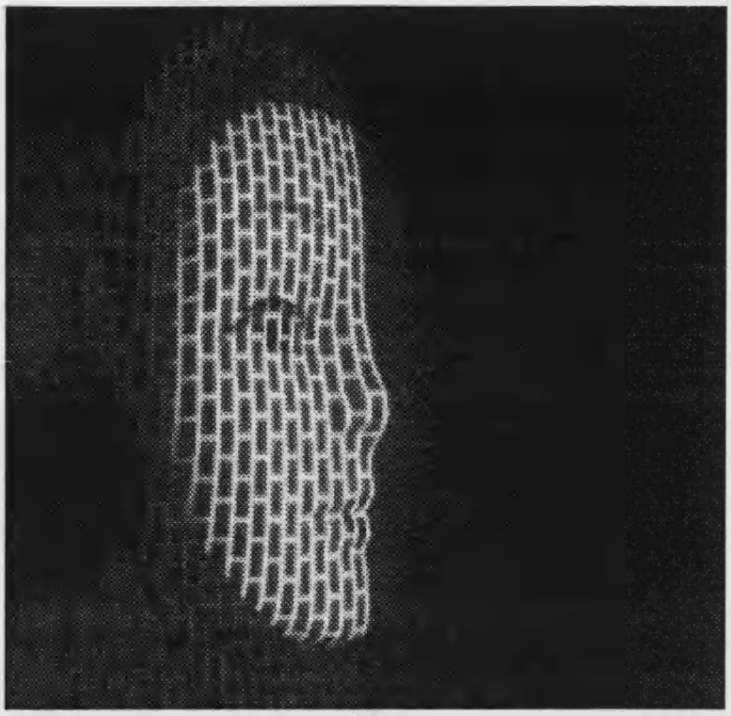


Figure 2.7: Reference lines on the middle of a model face, the head, ears and neck were covered

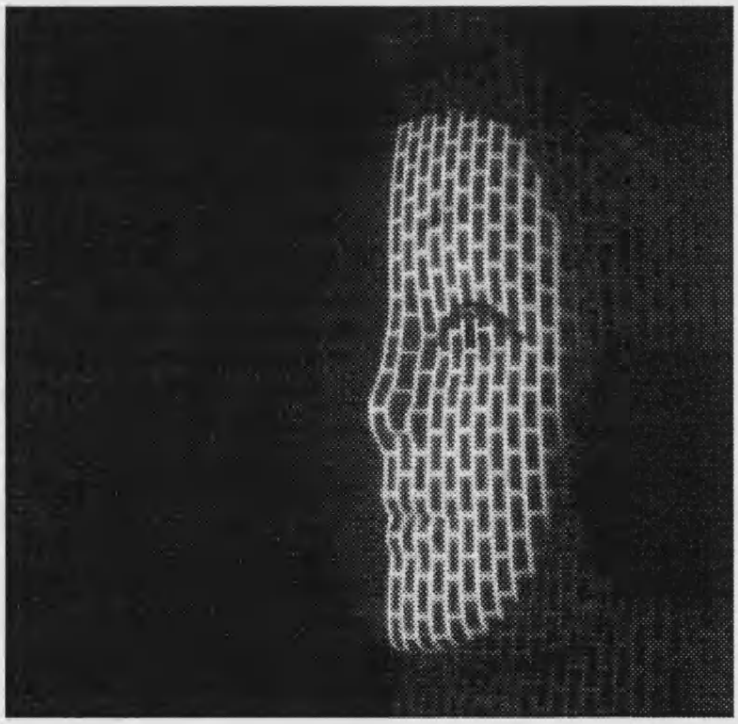
After focusing and levelling the projector and adjustment of the patient position by the operator, the system is ready to obtain the images from each side of the face. The angles between the cameras and projector must be the same, this is a necessary condition for a symmetrical system and is usually between 30°-45°. The images from the left and right cameras are presented on the left and right halves of the monitor screen, positioned at the centre to avoid distortion at the screen edges. The line on the profile of each image must be matched with the labelled centre line on the monitor.

The next step is to adjust the camera and projector distance (d_c, d_p) from the base plane. The first one is done by considering the field of view, Appendix C, and lens distortion for the camera. When the subject is close to the camera, barrel distortion will occur on the image. To have a linear image from the face, the optimum condition was found and then the maximum area on the monitor for monitoring this image, was determined. The distance of the projector from the face is adjusted by considering the number of projected vertical lines on the face. To obtain an image of the whole face the projector needs to be approximately 70-75 cm away.

The images from each side of a face have been captured and processed separately and then connected together to produce a complete 3-D reconstructed image as explained later. Both images have been digitised at 512 by 512 pixels resolution and 256 grey levels as shown in Figure 2.8. To have a perfect image from the face for the post processing stages, the head, ears and neck should be covered by a black cover.



(a)



(b)

Figure 2.8: Captured images from a model face, (a) left side and (b) right side of the observed face.

The optical system constant parameters and set-up information for capturing the images from a model face is shown in Table 2.1 and Table 2.2. Measurements are derived from approximate manual measurements, taken to ±1 mm tolerance.

f_p	90 mm
φ_p	25 mm
d_W	0.1 mm
d_B	0.7 mm
f_c	4 mm

Table 2.1: The optical system constant parameters

Distance of the projector to base plane (d_p)	730 mm
Distance of the cameras to base plane (d_c)	340 mm
Distance between the camera and projector (d_{cp})	240 mm
Camera-projector angle (θ)	35°
Camera aperture ($D = f_c / N$)	f_c/10
Image size (pixels)	512 × 512

Table 2.2: Set-up information for capturing two sides of the model face.

2.7 Conclusion

The principle of the structured light theory was introduced to recover the height data from the object's surface using triangulation as the basic technique. The methodology of the camera and projector calibration permits the calculation of height of the labelled points on the object's surface.

A flexible and reliable system has been designed for the optical set-up. A person sits in front of the optical set-up and the operator, by using the preview facility, controls the system. Adjusting and levelling the optic set-up should be done before capturing the images.

The high resolution hardware is the necessary key for transferring the data into the computer memory and of course for the post processing. The requirements of the system such as speed and accuracy were considered and suitable hardware designed and implemented.

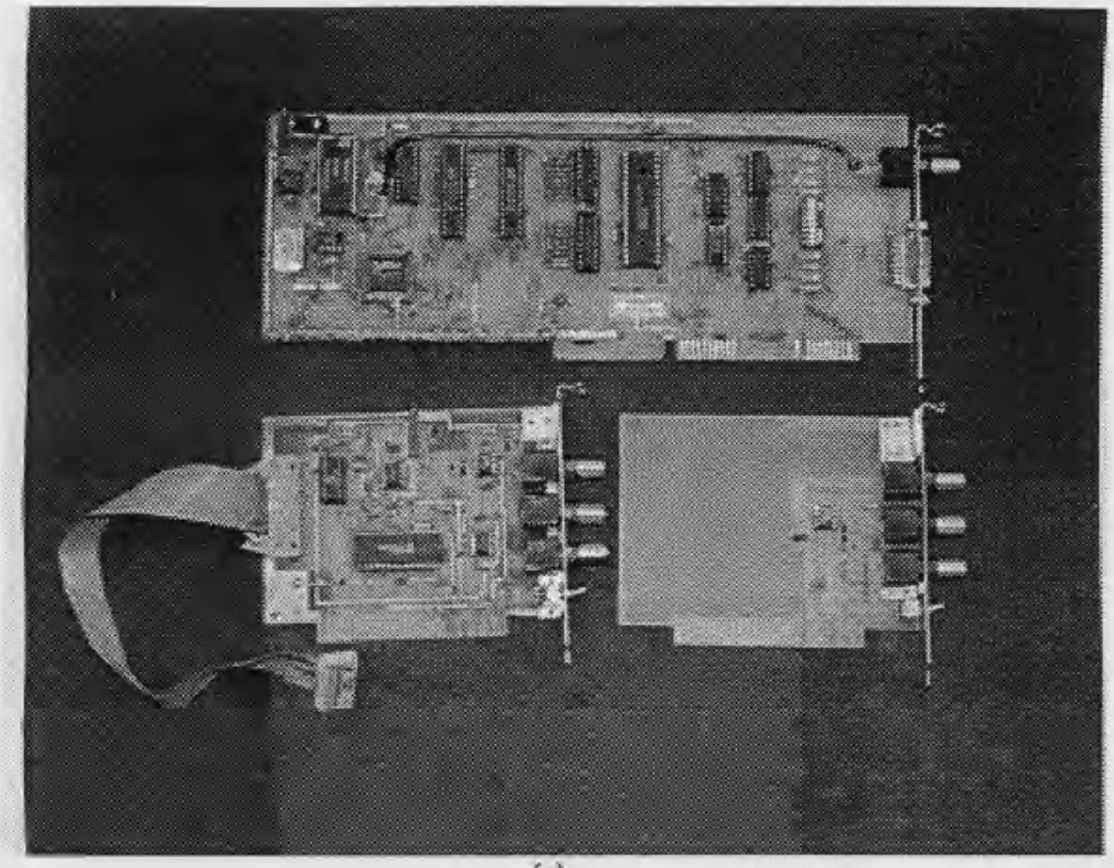
Chapter 3

Hardware Overview

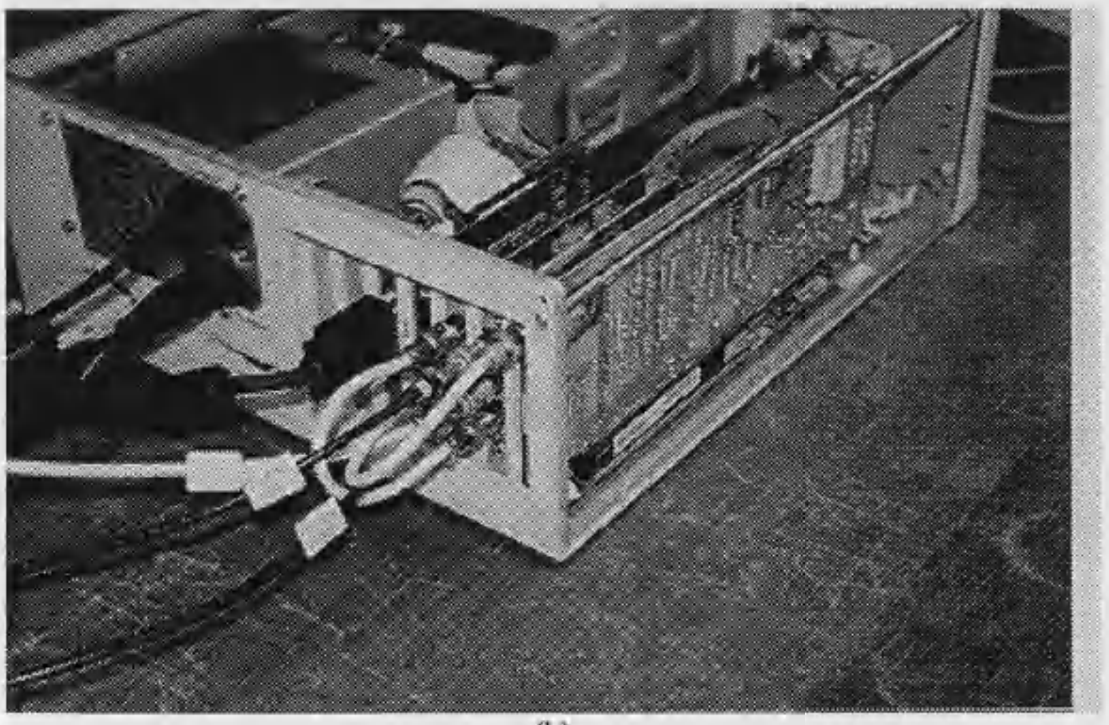
This chapter expands on the requirement list for a structured light system discussed in the previous chapter. A real-time frame-store is developed and together with two CCD cameras, a cost-effective, accurate system for capturing the structured light images into a computer is realised.

The frame-store includes the frame-grabber, frame-preview and video multiplexer cards that can be seen in Figure 3.1. These three cards are connected directly to the PC expansion bus connector and the video cameras can be plugged into them. The advantages of using PCB cards for the frame-store are:

- A ground plane can be used to protect the analogue signals.
- Noise is reduced by using the shortest path to connect signals to the computer busses.



(a)



(b)

Figure 3.1: (a) Frame store cards, (top) frame-grabber, (left) frame-preview and (right) video multiplexer (b) the connections of the cameras and monitor to the cards.

The main block diagram of the hardware is shown in Figure 3.2. The video signals from the two cameras are applied to the video multiplexer (1), to be selected by the operator sequentially. Then the video splitter buffers the selected signal to feed the composite video signal to sync separator and the A/D converter and also the video multiplexer (2). The sync separator extracts the synchronisation signal from the analogue video signal, and also a flag for odd and even screens from camera 1 and camera 2. The other block which uses the video signal is the analogue to digital converter with anti aliasing filter and AGC controller. This converter is also supplied with the synchronisation signals to set the gain for the analogue signal. It produces an eight bit digital output that is buffered and sent to the SRAMs.

The Lattice PLD is programmed via the computer before use. It receives the synchronisation signals and commands from the PC and produces control signals for digitisation and downloading, and provides addresses for the SRAM's. It also produces a clock signal for the system.

For the preview facility, a video signal from the camera via video multiplexer (2) or the digitised signal from the ADC via the DAC, is selected by operator. By selecting the reconstructed signal from the DAC, the ADC gain for compensating the camera intensity can be adjusted. This gain is adjusted manually (Manual Gain Control) so that the A/D converter can adapt itself to the output level of the various video sources and ensures that the best image with the necessary resolution from the subject is captured.

After the grabbing mode, all of the information in the SRAMs is ready for transferring to the computer's memory. This is done by the I/O controller chip. During the downloading mode data is taken from the output buffers of the SRAM's and presented onto the data bus. Besides the transferring of data, all of the commands to or from the PC are connected by the I/O controller to the circuit.

For activating the system from the computer, a base address is defined by the address decoder circuit. The complete circuit diagrams of the frame-store are shown in Figures 3.7-3.10. The elements of the frame-store will be described in the following subchapters.

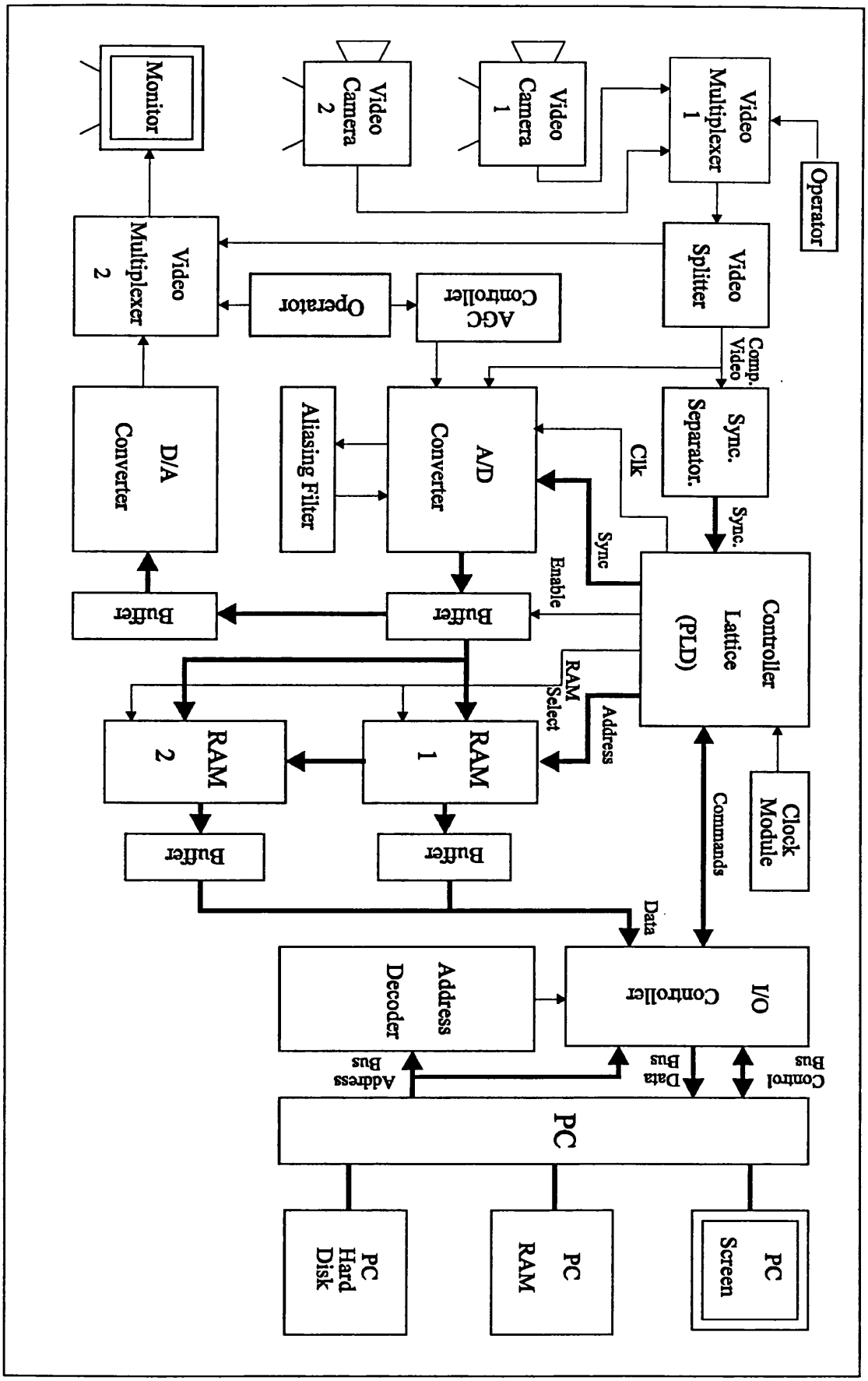


Figure 3.2: Main block diagram of the frame-store

3.1 Video Active Splitter (Figure 3.7)

The CCD camera presents the output at a 75 Ω BNC. It gives out a standard video signal that includes line and screen synchronisation pulses and the analogue voltage representing luminance of the image associated with the x position as explained in appendix C. The video signal from the camera has to be connected the sync separator, ADC and monitor via video multiplexer 2. The VAS (U13) is used to prevent distortion and reflection in the video signal caused by mismatching and overloading of these three loads. The MAX457 [57], from MAXIM Integrated company, contains two unity gain video amplifiers that are capable of driving 75Ω loads with a -3db bandwidth of 70 MHz.

3.2 Sync Separator (Figure 3.6)

The easiest way to extract the synchronisation signals is to use a commercial synchronisation separator supplied by Elantec (U9), the EL457CN [58]. The synchronisation outputs of this IC as shown in Figure 3.3 are all active low and are mainly used by the Lattice PLD to steer the digitisation. The Csync output is active during the line synchronisation of the video signal and Vsync output is active during the frame synchronisation. This IC also generates an output to show whether the odd or the even frame is processed and sets the back porch output during the black level of the video signal.

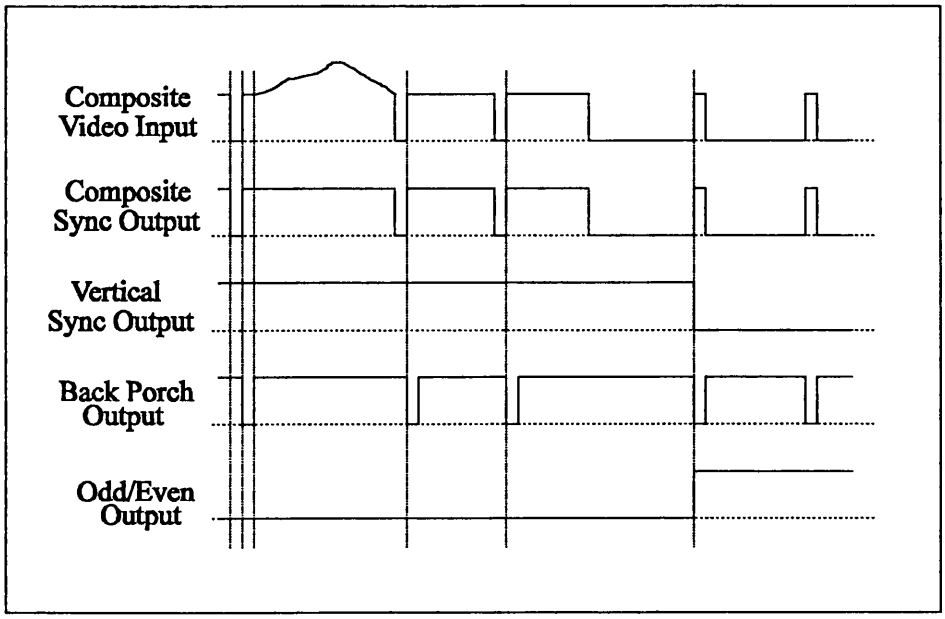


Figure 3.3: Output signals of the synchronisation separator

3.3 Analogue to Digital Converter (Figure 3.5)

For the given requirements the Philips TDA8708 [59] is chosen because of its fast sampling rate of up to 30 MHz, its accuracy and the clamp and automatic gain control (U1). The output of the video splitter is applied to the input BNC1 and is AC coupled to one of the three ADC inputs. The desired input is selected by the state of the SEL0 and SEL1 inputs which in turn are controlled by computer. It digitises the input signal of around 1v to an eight bit binary output.

Once the input has been selected it is passed through an anti-aliasing filter so that any colour burst component in the video signal does not affect the digitisation of the luminance. The anti-aliasing implementation is done by the internal operational amplifier and external passive components. An ADC obviously needs a reference voltage or an input gain to adjust the range of the digital output voltage to the range of the analogue input voltage. All the eight output bits should be used for best resolution of the digitised signal. This chip uses an automatic gain control and a clamp level input to amplify the input signal so that the full digital output range is used.

TDA8708 has two modes for working. In configuration mode 1 the video signal is too weak and therefore the gain of AGC amplifier has not reached the optimum value. Mode 2 occurs when the Gate A input of the ADC is connected to an active high line synchronisation pulse and the Gate B input is connected to an active high back porch signal. In this mode when Gate A is high, the output of the AGC is clamped to digital code 0 and when Gate B is high, the output is clamped to digital code 64. Thus if Gate A is activated during the sync level and Gate B is activated during the backporch interval, the AGC will process the video signal such that the black level is fixed to digital code 64, the peak of white level will never exceed digital code 240 and all sync information will be below digital code 64. The clamp level control works in the following way: the black level digital comparator is active during the backporch pulse at the Gate B input, so the clamp capacitor is charged or discharged to adjust the digital output to 64.

Two facilities are considered for AGC in the AGC controlling circuit. First, it can be adjusted in automatic gain by choosing a recommended capacitor for the AGC pin. TDA controls the charge or discharge current of this capacitor according to the signal level. The voltage across this capacitor controls the gain of analogue amplifier. Another facility in this implementation is, the AGC pin can be connected to a variable voltage reference to allow the user to manually set the gain level. With the preview facility, the user may vary the gain setting with the potentiometer until the result is acceptable.

The ADC operates synchronously with a five volt logic clock applied to pin 5. With reference to the data sheet, the ADC samples the output of the AGC on the rising edge and produces a valid digital code about 28ns (max.) later. The output of the ADC is buffered by U3 before it is presented onto the main frame-store tri-state data bus.

3.4 Screen Memory (Figure 3.5)

The screen memory was implemented using SRAM chips (U4 and U5), which although more expensive than DRAM's, have faster access times and produce no problem with refresh timing. One screen include, 512 lines with 512 pixels and each pixel contains eight grey levels, so two 128 Kbyte SRAMs are chosen to store the odd and even frames.

The primary concern when designing a frame-store memory is that of access speed. The necessary access time for SRAMs is controlled by the camera sample time. One screen is captured by the camera at 40msec and has 625 lines and therefore one line takes.

$$T_{one\_line} = T_{one\_screen} / N_{(camera\_line\_numbers)} = 40m\text{sec}/\ 625 = 64\mu\text{sec} \qquad \text{Equation 3.1}$$

12μsec of that time is used for the synchronisation signal and setting the black level and only 52μsec remains for data. The sampling time for each pixel is calculated by knowing the number of pixels for each line,

$$T_{sample} = T_{data} / M_{(Pixel\_Numbers/Line)} = 52 / 512 = 100\ nsec \qquad \text{Equation 3.2}$$

However, time should be allowed for propagation delays through the ADC, data buffers and the delay times of the control logic. The ADC has 28nsec delay time and the output buffer has 12nsec so the access time of SRAMs should be,

$$T_{SRAM} < T_{sample} - T_{delays} = 100n\text{sec} - 28n\text{sec} - 12n\text{sec}$$
$$T_{sram} < 60n\text{sec} \qquad \text{Equation 3.3}$$

This is the maximum time that may be allowed for an access assuming no other delays in the circuit. The SRAMs with 20nsec access time (IS61C1024) supplied by Integrated Silicon Solution Inc. [60] were chosen.

All addresses and WE and OE signals for the SRAM's are generated by the Lattice PLD for both digitising and downloading, and each SRAM is selected alternately by Csx pulses during the odd or even screen.

3.5 Lattice PLD (Figure 3.6)

For operating the frame-store, a programmable logic device was chosen (U11), Lattice ispLSI 1016 [61 , 62], instead of a microcontroller because of the ease of programming and structure as shown in Figure 3.6. This device can be reprogrammed on board with a special plug connected to the PC in about 10 sec, therefore it makes the debugging of the hardware easier than an EPROM. Also the PLD can operate at a 120 MHz clock frequency which is appropriate for the number of pixels per sampled line. The logic blocks on the chip are programmed from an integrated environment like a normal software compiler. The logic is entered either by putting in equations or by using macros for logic blocks, i.e. counters or flip flops. After the logic has been verified and routed to the given resources, the PLD is programmed by connecting the parallel port of the PC to a plug mounted on the board with a special cable supplied with the software package. This plug is wired to some special inputs of the PLD which have a dual function, ordinary pin and programming pin, to allow programming of the device in situ.

3.5.1 General Functions

The PLD can be considered as a counter for counting lines, pixels and delays. The line counter can count 256 lines for each screen, odd or even. The pixel counter needs nine bits to count 512 pixels per line. Also the pixel counter is used as a delay counter for a proper initialisation of sampling.

In general, the Lattice PLD should control the whole hardware on the board when it is triggered by the commands for digitising or downloading from the PC (SAMPLE input). S3 is a flag for the Lattice if it is waiting for a download command or for a command to start digitisation, and OE is for the data synchronisation with the PC.

The active low synchronisation signals are put to the Lattice from the Sync Separator. The CLK_20 input is connected to an oscillator module and the active low POWER_ON_RESET signal resets all internal flip flops at the beginning of power on. The main outputs are the address outputs ADDRxx, the active low OE and WE signals, RAM select signals (RAM1, RAM2) for the appropriate RAMs and 10 MHz clock output for connection to the ADC. BPORCH and CSYNCH are inverted into the PLD and then go to the ADC as well.

3.5.2 State Machine

The state machine is the main part of the PLD and can be divided into two parts, the first part for the digitisation of the picture and writing it to the SRAMs and the second part for reading the picture from the SRAMs and transferring it to the PC. All of the control sequences are done according to the state machine.

The user friendly graphics environment permits to the user to select the grab mode by pressing (G) on the keyboard. With pressing this button, the first step of the process starts by digitising the image of a single frame for one camera and moving the digitised data from ADC to the SRAMs. The digitised data from each screen, odd or even, is positioned to the suitable SRAM, odd or even, by the enabling signals for each SRAM which are activated by the PLD.

After capturing a frame and writing the digitised data into the SRAMs, the second step of the process starts by reading the data from SRAMs and transferring it to the PC bus. The odd and even data are transferred into the I/O controller and from there to the computer. Then the software will put these two screens, odd and even, together to display a full frame on the computer monitor. For capturing the image from the second camera, the grab mode must be repeated. The program and other necessary information about the PLD and state machines are explained in Appendix C.

3.6 Image Preview Hardware (Figure 3.7)

The purpose of this hardware is to provides very fast and independent preview facility of the face before grabbing it. The circuit to do this is shown in Figure 3.7, it consists of a digital to analogue converter U14 (DAC-08), the video multiplexer U16 (DG538ADJ) and low noise amplifiers U15 and U17 (NE5534) [63]. The digitised image after ADC is fed directly to the buffer and DAC. The reconstructed video signal is put to the video multiplexer (2), another input for this chip is the video signal from camera and splitter. By activating the channel select of the video multiplexer, the result from DAC or camera goes to the video out connector. The user may adjust the gain of the ADC by switching the monitor to preview facility before grabbing the image.

3.7 Clock Module (Figure 3.6)

The time basis for the synchronisation digital parts of the board is set with a standard oscillator module U10 [64]. The clock frequency is chosen to 20 MHz to achieve the appropriate sampling rate. The clock signal is fed to the Lattice PLD to be processed.

3.8 I/O Controller (Figure 3.8)

The Intel 8255A (U23) Programmable Peripheral Interface (PPI) [65] is a general purpose I/O device which provides three 8-bit I/O ports (port A, B and C), The commands are given from the PC program and put to the Lattice by the 8255 and then the data and synchronisation signals from the digitiser transferred to the PC from this unit as shown in Figure 3.4.

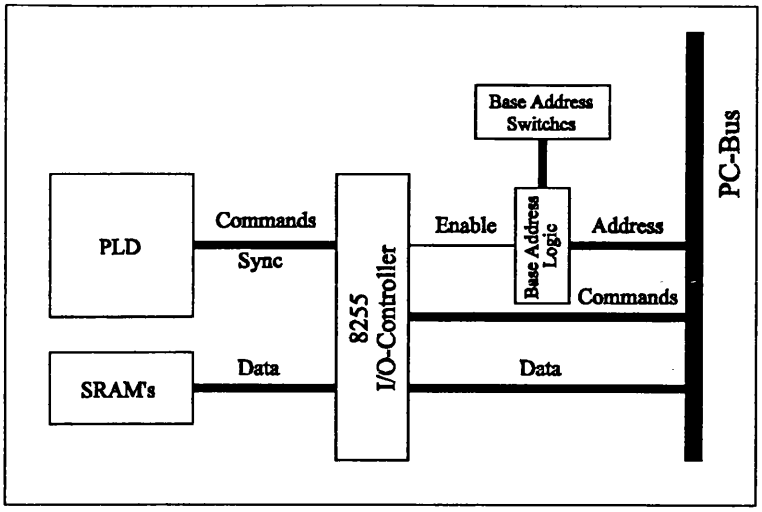


Figure 3.4 : The Interfacing between the frame-grabber and PC bus

The operation of the I/O ports is controlled by the format of the 8-bit word written to the control register, located at address (Base + 3). For our application, the control word is chosen 0x93 so that the 8255 will work in mode 0, which allows simple input-output operations without handshaking. With this configuration the port A and port B are inputs, the lower nibble of port C is input and the upper nibble of port C is output.

The APIN byte and BPIN byte are used as the data input from the hardware to the PC, so they are directly wired to the output buffers of the SRAMs, each one transports the odd and even byte to the PC in the downloading process. The OE bit for the downloading synchronisation is wired to the LSB of the lower nibble of the CPIN byte, to get it to the PC. This nibble has three spare bits, so the state bit S3 is taken to the PC for debugging information. The command bit SAMPLE for the Lattice from the PC is connected to the LSB of the higher nibble of the CPIN byte. The other three bits of that nibble can be used for new functions.

3.9 Address Decoder (Figure 3.8)

The I/O controller can read or write data when it is given an enable signal derived by the base address logic. This enable signal is set when the access address on the PC bus is the same as the address is set by the DIP switches on the card (A9.....A2). The other two address lines (A1-A0) are supplied for selecting the registers in the 8255 controller. Logic gates (U18, U19, U20, U21 and U22) were chosen for accessing the base address as shown in Figure 3.8. Due to the switches on the board, the base address can be set anywhere but it is recommended that to be set on 0x300 because this is the defined standard address for any prototype board (Table 3.1).

0x300	8 bit data word input from I/O controller (APIN)
0x301	8 bit data word input from I/O controller (BPIN)
0x302	4 bit command output (MSN), 4 bit sync input (LSN)
0x303	8 bit control word (0x93)

Table 3.1: The access address in virtual memory

3.10 Power Supply

The available power for additional expansion cards on the PC depends up on the rating of the system power supply, the requirements of the mother board, and the demands of other adapter cards which may be fitted. The recommended current limit for each designed expansion card is shown in Table 3.2.

Voltage Rail	Connection	Maximum Current
+5	B3 and B29	1.5 A
-5	B5	100 mA
+12	B9	500 mA
-12	B7	100 mA

Table 3.2: Recommended current limit for each expansion card

Where several adapter cards are fitted, the current demand for each supply rail should be estimated and the total power requirements calculated. It is clear that the total demand should not exceed the spare capacity rating of the system power supply. the maximum currents of each voltage were calculated using the data sheet values for the frame-grabber and frame-preview cards as shown in Table 3.3.

Supply voltage (Frame-grabber)	Measured current
+5	300 mA

Supply voltage (Frame preview)	Measured current
+5	45 mA
-5	50 mA
+12	20 mA
-12	15 mA

Table 3.3: Measured currents for the frame-grabber and frame preview cards

3.11 Program Access to Data

To make the use of the frame-store as easy as possible a program was written to read the data from the card, present it to the PC bus and put it to the screen as soon as possible.

The data and commands at the explained address (0x300) cannot be accessed with a pointer because those addresses are in the virtual memory area for input and output devices [66]. Therefore reading or writing from or to those addresses has to be done with the C-functions inport() and outport() that used the PC to get the data in from the card to the PC memory.

The C program has to wait for the OE synchronisation pulse to read the next two bytes after giving the command for downloading to the card. This is done in a while loop that waits as long as OE is high, indicating not active. When OE is changed to 0, new data is read at APIN and BPIN. After reading the odd and even byte for one pixel at the APIN and BPIN port, they have to be assigned to the picture data at the proper

places. The odd and even screens, each of 256×512 pixels, are put together to achieve a 512×512 pixels picture in the PC memory.

To write the data bytes from the card to the PC without any handshake and to prevent any interrupt problems during the data transfer, the interrupts are disabled at the beginning of the program and have to be enabled again after the downloading routine in the C program.

As mentioned previously the ADC (TDA8708) sets the black level to 64, therefore the program function _extend() brings the data range from 64 for black and 255 for white to 0 for black and 255 for white. This is done by subtracting the value for the old black level (64) and then multiplying the pixel value by 1.333 to obtain the full range again.

3.12 Process Time

As mentioned previously the two main parts of the timing are the digitising time and the downloading time; the digitising time is much shorter than the downloading time because of the structure of the PC bus. The hardware timing limits are set by the 8255 I/O controller which has a data cycle time of about 850 nsec for each byte pair, but the bus of the PC is too slow to transport the data to the PC memory in this time. Therefore the whole system has to be slowed down for downloading, resulting is a long delay loop on the PLD. The safe downloading of the two bytes data needs about 130 cycles of 100 nsec each. Thus the downloading of the whole image needs:

$$T_{process} = Data_{SRAM} \cdot n_{cycle} \cdot T_{cycle} = 128kbyte \times 130 \times 100n\,\mathrm{sec} = 1.66\,\mathrm{sec} \qquad \text{Equation 3.4}$$

One useful technique for reducing this time is to use a DMA controller instead of the I/O controller for transferring the data from the frame-store to the computer.

3.13 Conclusion

The implemented hardware, frame-grabber, frame preview and video multiplexer, was described in detail by dissecting the vital components. Also the additional information for implementing the system such as the timing and programming was introduced.

The merits of the system have been compared with commercial hardware by considering the following points, and a reasonable figure is obtained.

- Image resolution
- Data access time
- Noise immunity
- Cost

The quality of the captured images is good enough for using in different applications when an accurate image from a surface is desirable. This aspect helps us to reduce the amount of the enhancement on the captured image, which will be discussed later in chapter 4.

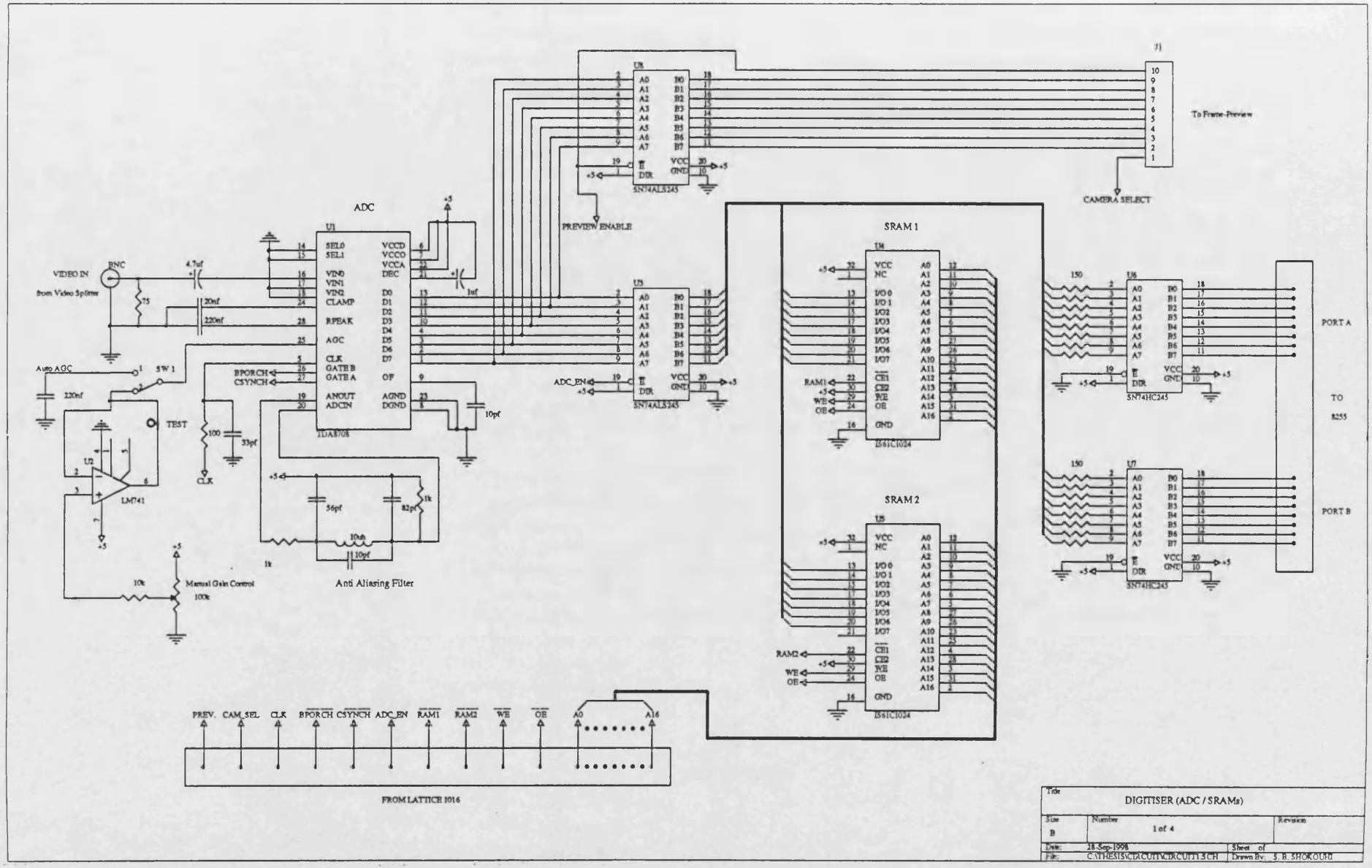


Figure 3.5: Digitiser circuit diagram (The ADC, SRAM's and Buffers)

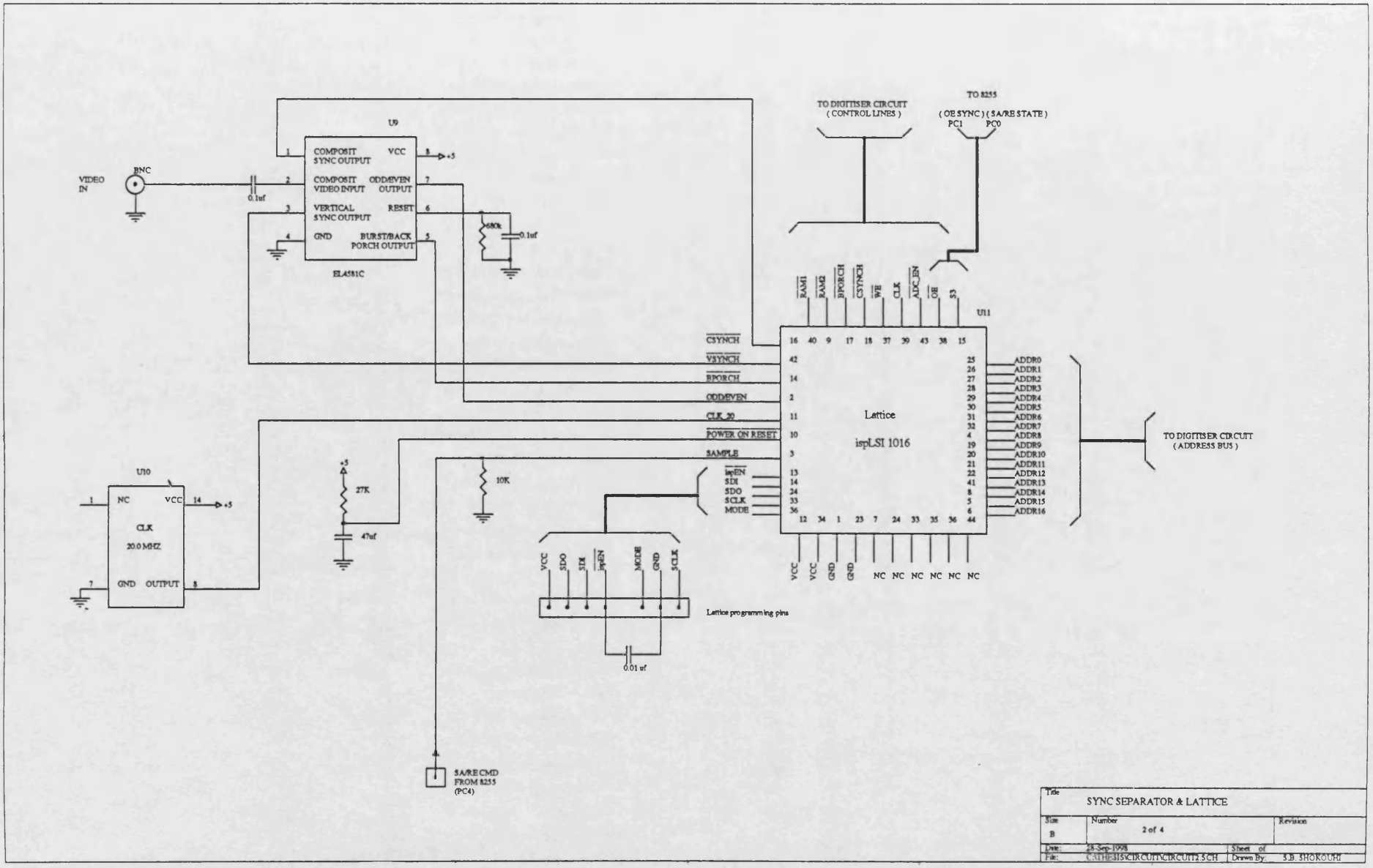


Figure 3.6: Controller circuit diagram (Lattice PLD, Sync Separator and Clock)

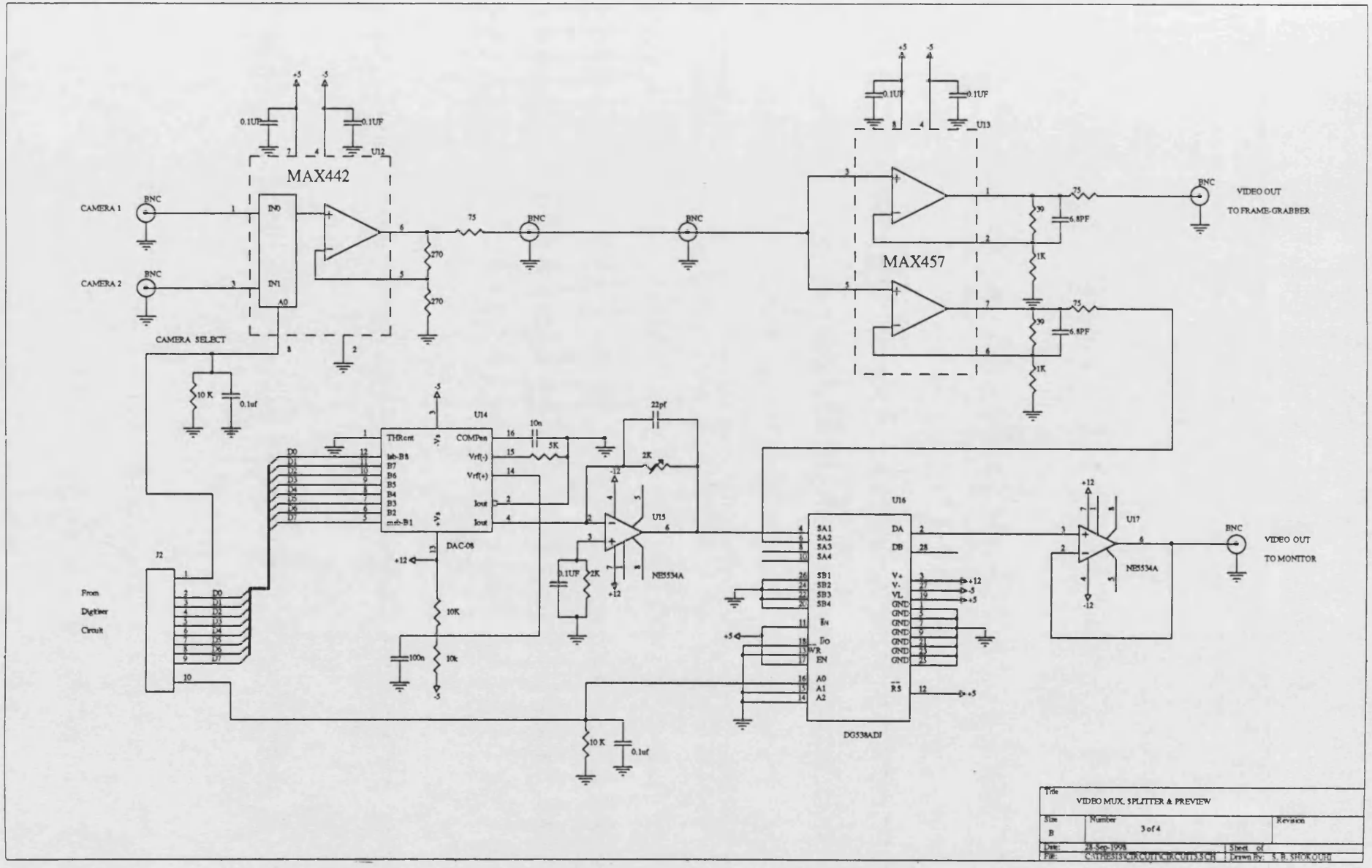


Figure 3.7: The Preview and Video Splitter circuit diagram

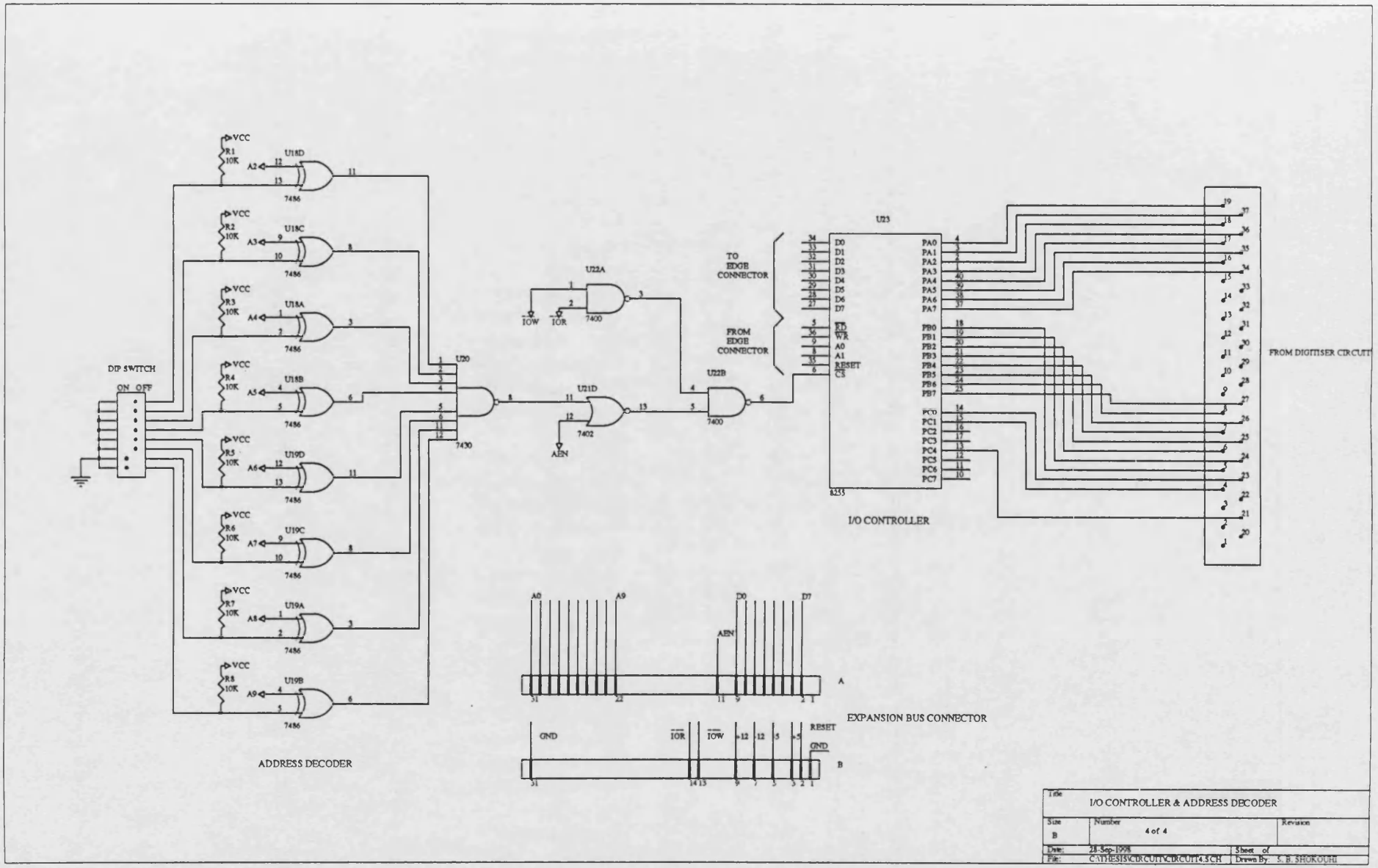


Figure 3.8: The I/O Controller and Address decoder circuit diagram

Chapter 4

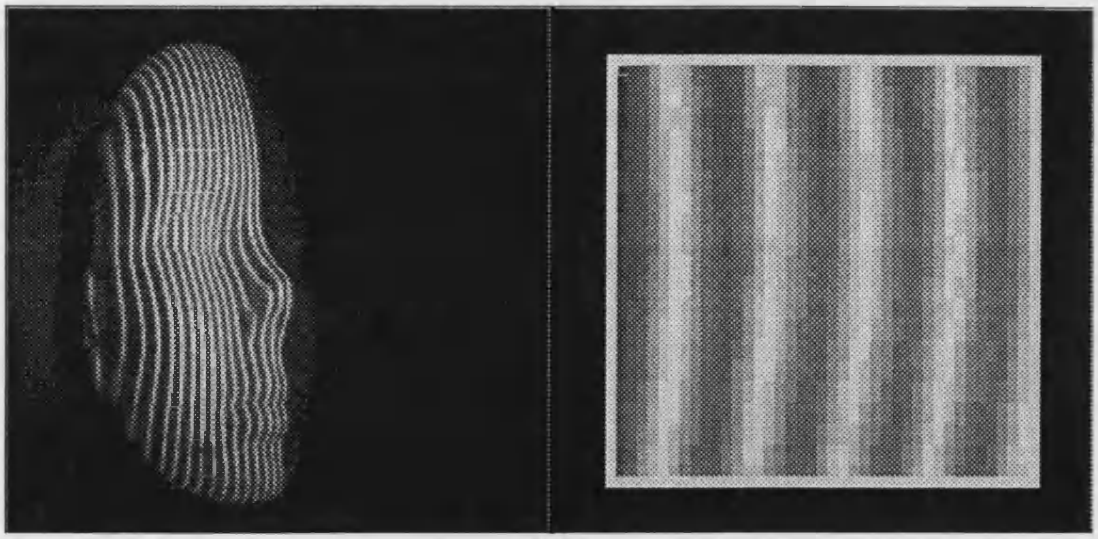
Image Processing

The use of white light as the structured light source rather than a laser avoids the risk of damage to the eye. However, because of the poorer contrast of the white light stripes, and because the projected white light can not be simultaneously in sharp focus over the entire surface, sophisticated image processing is necessary to produce a satisfactory image. The goal of the image processing stage is to generate facial lines without discontinuity and having single pixel thickness, by employing two main stages; feature extraction and feature interpretation, and these are described in this chapter and chapter 5.

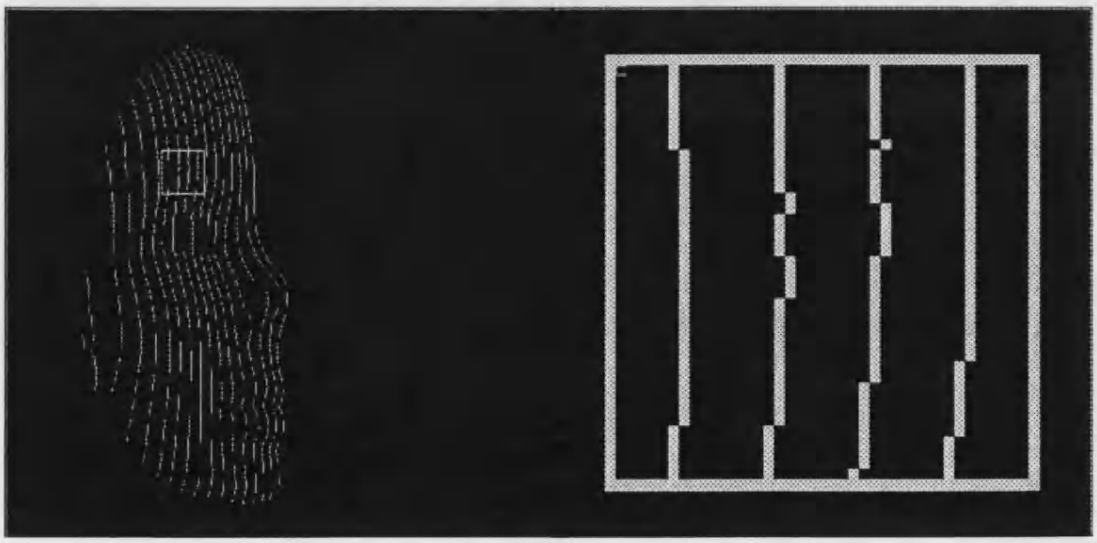
The image processing stages must be applied separately to both images of the sides of a face. The image processing algorithms for the face are developed in two groups, the left and right group algorithms, to produce separate processing for each side. Each group includes the general purpose feature extraction and also unique algorithms for feature interpretation. Most algorithms for implementing the feature interpretation are written in a symmetrical manner. For example when the tracing algorithm, as explained in Chapter 5, scans the image from the right to left for the left side image, the symmetrical process will scan from the left to right for the right side image. All of the image processing algorithms for feature extraction and feature interpretation stages at this thesis are explained for the left side of the observed faces such as shown in Figure 4.1.

4. Feature Extraction

The aim of feature extraction is to detect and enhance the structured light facial contours to yield a representation (feature) which is more useful in our interpretation of that image. A feature in our structured light system is a point determined to one pixel accuracy lying on a grid line which presents a local maximum in intensity. The line segments in Figure 4.1 from a parallel stripe pattern show the intensity information in a structured light image, together with the corresponding detected features according to these definitions.



(a)



(b)

Figure 4.1: (a) Intensity information from a grey level structured light image, (b) and corresponding detected features after feature extraction.

The feature extraction phase is blind to the exact nature of structured light images. It does not make any distinction between features which are caused by artefacts in the image and those which are directly caused by structured lighting. It is the task of the feature extraction described in this chapter to find the location of features in the image. The task of feature interpretation, described in the next chapter, includes the rejection of those features which are not directly caused by light stripes in the original structured light image.

The procedures used to automatically produce the extracted and enhanced image are shown in Figure 4.2. This software translates the image, with its real world problems such as poor contrast, uneven illumination and limited quantisation, into data that the feature interpretation will work with.

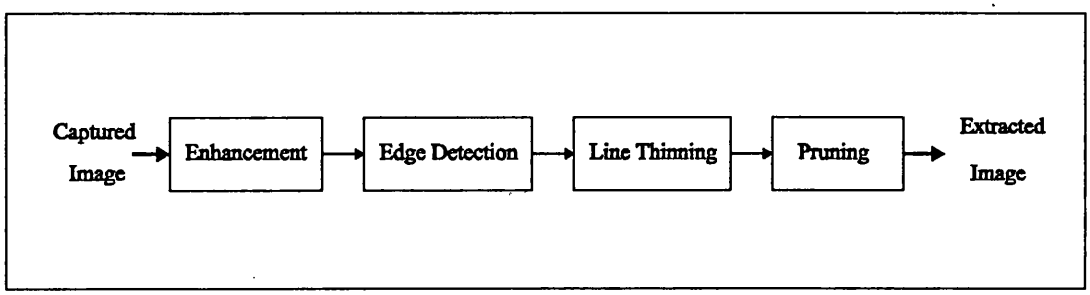


Figure 4.2: Auto feature extraction stages

The four basic processes introduced in feature extraction are:

1. Enhancement
2. Edge detection
3. Line thinning
4. Pruning

A number of algorithms were considered for each of the four basic processes and then the most suitable one was selected for each process. Filtering algorithms were applied on the captured image to enhance it, then the edge detection algorithms followed by thresholding were used to form the binary stripes. Finally, line thinning methodology was introduced to generate one pixel thickness lines. To remove some noisy pixels or lines from the resulting skeleton, pruning algorithms were used which proved to be most suitable for this approach.

Before covering the approach taken in more detail, it is best to summarise some of the characteristics and difficulties associated with detecting features in the structured light images:

1. The structured light lines includes a wide variety of pixel numbers as shown in Figure 4.1 (a). The stripe widths vary from 1 to 10 pixels, which in itself is sufficient to cause feature extraction problems.
2. The intensities of the stripes can vary widely depending on the surface gradient.
3. Despite correcting the structured light slides, as described in chapter 2, intensities will have some local variation according to the colour of the surface that they are projected on.

Thus, the extraction scheme is required to cope well with the resolution problem, and to be able to detect features where intensity and contrast are highly variable (which is the case in structured light images taken from complex surfaces such as the human face).

4.1 Enhancement

The principal objective of enhancement is to process an image so that the result is more suitable than the original for our specific application. A wide variety of techniques have evolved for enhancing images in the spatial or frequency domain. Image enhancement is a rather subjective matter because what is 'more suitable' depends on the type of details and contrasts in the image that the user is hoping to acquire.

The main step for achieving a reasonable image and reducing the noise problem is to find a suitable smoothing algorithm by using point or group operators. Three common methods, histogram equalisation, median filtering and Gaussian filtering, were all tested on the captured images, and the most suitable one selected as explained in the conclusion of this chapter. A test image from one side of a real face is used to show the result of the processing is shown in Figure 4.4.

The co-ordinates of an image on the screen have been defined as, x for horizontal axis (column) and y for vertical axis (row), thus for any image point (P) on the screen, the co-ordinates is $P(x_p, y_p)$. The image size is x_{max}=512 and y_{max}=495 as shown in Figure 4.3.

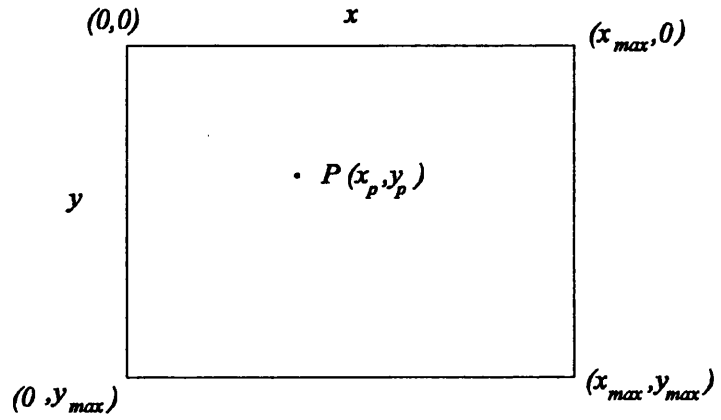


Figure 4.3: The screen co-ordinates

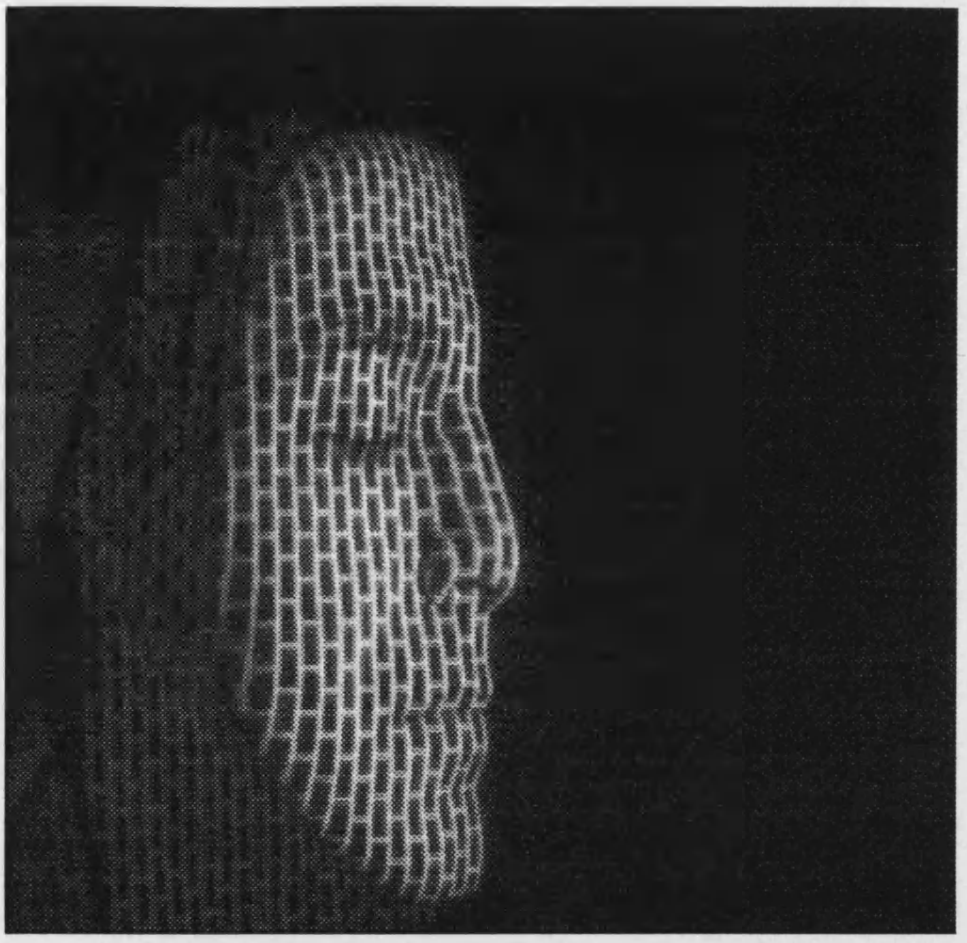


Figure 4.4: Captured image from the left side of a 'face'

4.1.1 Histogram Equalisation and Specification

To minimise the effect of the variations in contrast apparent in structured light, some form of histogram equalisation or specification may be used. The histogram of a digital image with grey levels in the range [*0, L-1*] is a discrete function $p(r_k) = \frac{n_k}{n}$, where r_k is the *k*th grey level, n_k is the number of pixels in the image with that grey level, *n* is the total number of pixels in the image, and *k=0, 1, 2, ..., L-1*.

A plot of this function for all values of k provides a global description of the appearance of an image. For example the histogram of a dark image would have more values of $p(r_k)$ biased toward 0 whereas a brighter image would have values tending toward *L-1*. The goal of any histogram equalisation process is therefore to improve the contrast by applying a transform of the form,

$$s = T(r)$$ Equation 4.1

which produces a grey level (s) for every input grey level (r), with the overall aim of achieving greater dynamic range. The histogram for the 'face' is shown in Figure 4.5.

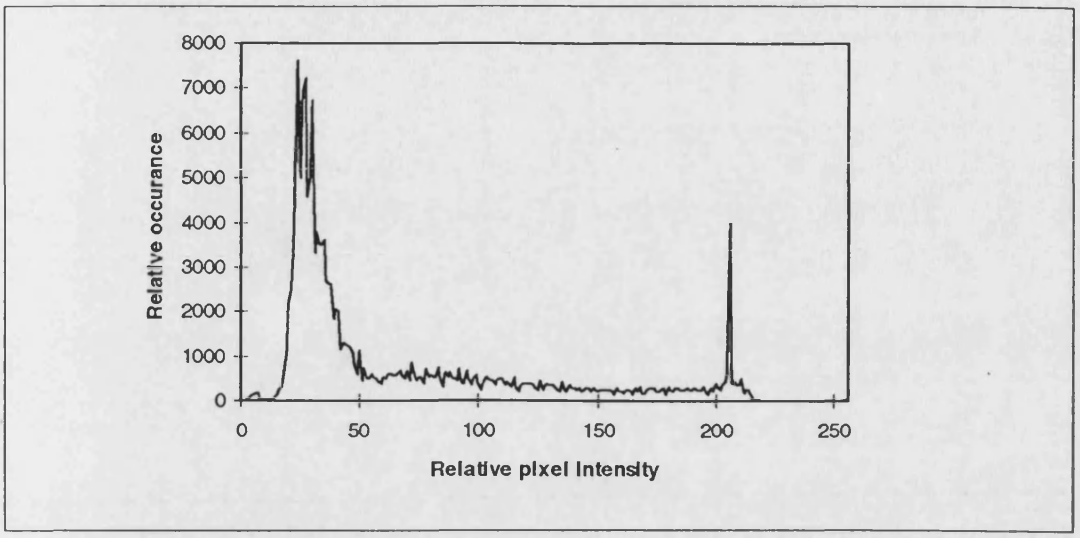


Figure 4.5: Grey levels histogram for the 'face'

4.1.1.1 Adaptive Histogram Equalisation

The histogram equalisation operation may be applied on either the whole image or smaller parts. Taking the image in smaller parts has the advantage that contrast variations in the image will be compensated for as then the process effectively becomes adaptive. Adaptive histogram equalisation [67] was defined by the equation 4.2,

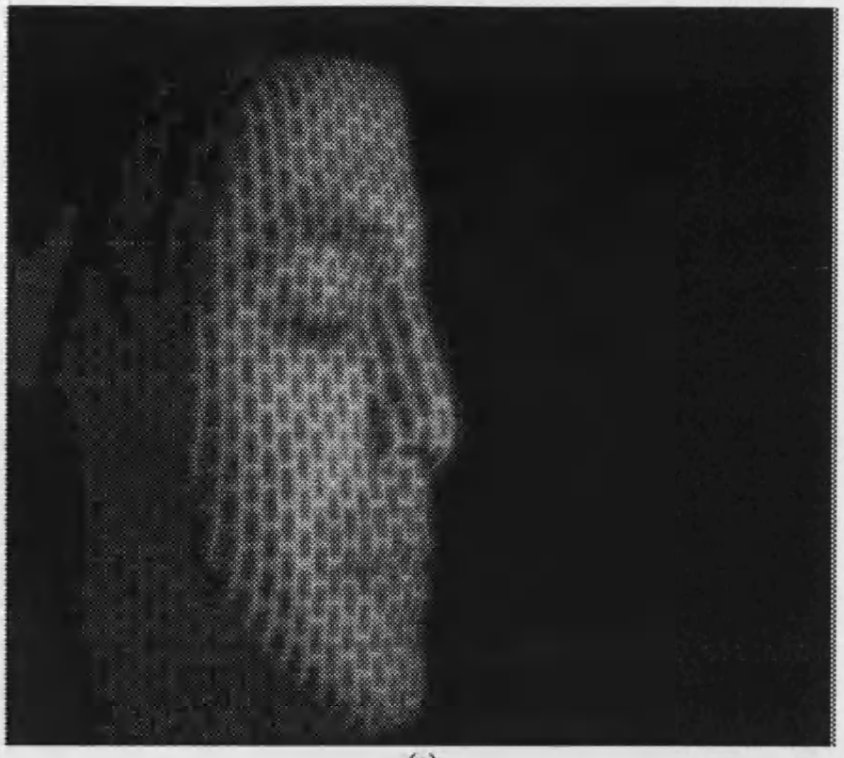
$$g(x,y) = A(x,y)\cdot[f(x,y) - m(x,y)] + m(x,y) \qquad \text{Equation 4.2}$$

where

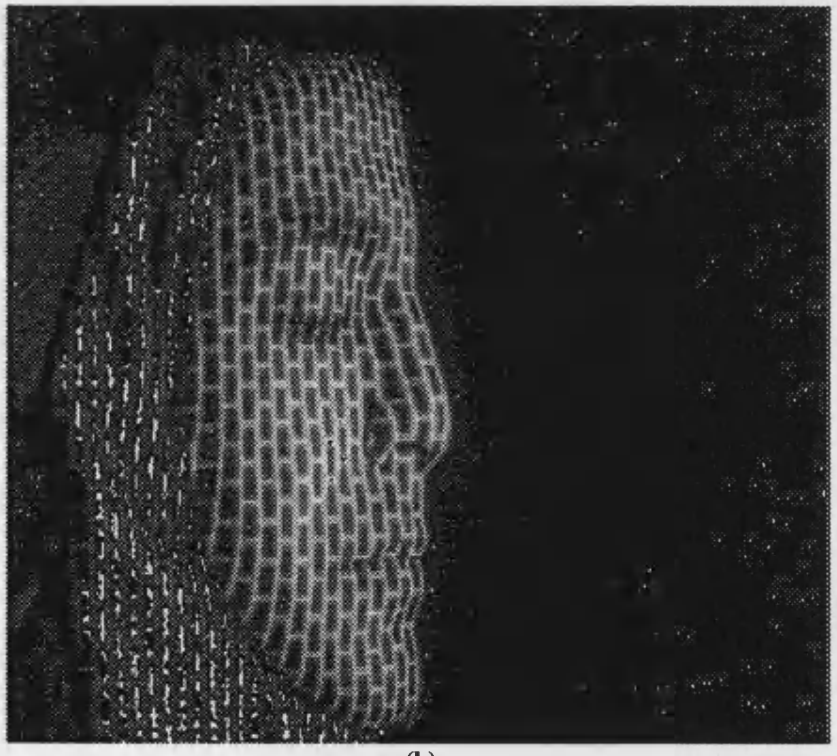
$$A(x,y) = k\cdot\frac{M}{\sigma(x,y)} \qquad 0<k<1 \qquad \text{Equation 4.3}$$

$m(x,y)$ and $\sigma(x,y)$ are the grey level mean and standard deviation computed in a neighbourhood centred at (x,y), M is the global mean of $f(x,y)$, and k is a constant in the range 0 to 1. The neighbourhood around $f(x,y)$ with which $A(x,y)$ and $m(x,y)$ are calculated is typically seven by seven pixels wide. In this region, a lower calculated $\sigma(x,y)$ will result in a higher gain being applied due to the inverse proportionality of equation 4.3 and so the areas of low contrast receive larger gains as shown in Figure 4.6.

However adaptive histogram equalisation techniques must be applied with care in the presence of unfavourable signal to noise ratios. If local histogram equalisation is performed on an area with low information content anyway, then the effect of applying the operation will be to amplify noise.



(a)



(b)

Figure 4.6: Face image after adaptive histogram equalisation performed at 11 by 11 area of detail with (a) k=0.3, and (b) k=0.8.

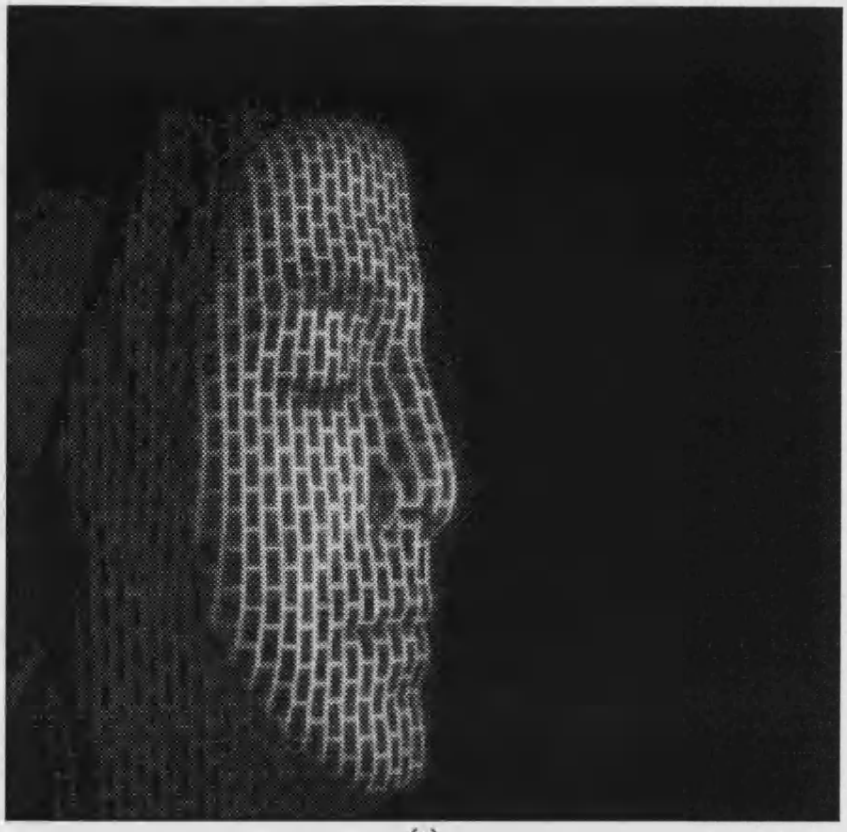
This problem can be seen in Figure 4.6 where although there appears to be a great overall improvement, many areas are now too noisy for lines to be extracted from them. One solution is histogram specification whereby the user sets the grey levels that the image is expanded to. In other words, instead of trying to achieve an evenly distributed histogram, the user determines how the histogram should look and the image is manipulated accordingly. For our application with the idea of automatic extraction, using the interactive histogram specifications is not acceptable.

4.1.2 Median Filtering

In this application the signal to noise ratio is frequently low and therefore any approach must cope with noise. The obvious approach would be to use a two dimensional implementation of a low pass filter, for example a Butterworth filter, in the frequency domain. However conventional low pass filtering smoothes over edges, since edges are the high frequency component of the picture, as well as the noise. The solution to this problem is to use the statistical tool of the median. The median, m, of a set of values is such that half the values in the set are less than m and half are greater than m.

The median is similar to an average, which is effectively what a low pass filter is, except that instead of computing the average of the neighbourhood we compute the median. The median filter is superior to the moving average filter in that is better at presenting sharp features while eliminating noise, especially impulsive noise.

In order to perform median filtering in the neighbourhood of a pixel, the pixel and its neighbours are sorted by value, the median determined and that value is assigned to the pixel. Ideally the median filtered image should be a noise free representation of the input image but the size of the neighbourhood area will affect the result in other ways. If the area is too large for a given image, sharp detail will gradually be 'spread out' giving rise to an oil-painting effect. The results of median filtering with two different window size is shown in Figure 4.7.



(a)

(b)

Figure 4.7: Median filtered face by (a) 3×3 and (b) 5×5 windows

4.1.3 Gaussian Filtering

Gaussian filtering, or blurring, is applied to smooth an image and so minimise the effect of quantisation noise. Smoothing is needed in many operations, especially derivative operative processes. A Gaussian filter is defined as shown in equation 4.4.

$$g(x,y) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2} * f(x,y) \qquad \text{Equation 4.4}$$

where f(x,y) is the input image
σ is the standard deviation of the Gaussian distribution
g(x,y) is the resultant Gaussian filtered image

This convolution may be implemented efficiently with two computational operations. Since the Gaussian kernel is separable, the convolution is performed in one direction and then in the orthogonal direction [68]. The second facility is that the Gaussian may be approximated by the iterated convolution against a discrete kernel. In one dimension, if a discrete signal *f(i)* is iteratively convolved against the three map kernel [1/4 1/2 1/4] *N* times, then the result is the same as the convolution,

$$g_N(i) = \sum_{k=-N}^{N} \frac{1}{2^{2N}} \binom{2N}{k+N} f(i+k) \qquad \text{Equation 4.5}$$

This is an approximation to convolution against a Gaussian with $\sigma = \sqrt{(N/2)}$. The approximation improves as *N* increases. In two dimensions the iterated convolution is performed *N* times against the three-by-three mask,

$$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \qquad \text{Equation 4.6}$$

This yields an approximation to the two-dimensional convolution against a Gaussian with $\sigma = \sqrt{(N/2)}$. *Davies* [69] investigates the accuracy of the 3×3 Gaussian kernel and derives a better result which is shown in equation 4.7.

$$\frac{1}{56}\begin{bmatrix} 3 & 7 & 3 \\ 7 & 16 & 7 \\ 3 & 7 & 3 \end{bmatrix} \qquad \text{Equation 4.7}$$

The smoothed image after applying this Gaussian filter, equation 4.7, is shown in Figure 4.8.

Figure 4.8: Face after Gaussian filtering

The different types of enhancement methods have been examined and tested for pre-processing of the captured image, but only two of them are recognised as the best for the application. The results of the median and Gaussian filter are very similar because of the good quality of the captured image. The Gaussian filter is also an integral part of many edge detection processes and so we shall favour its use over the median filter. The results of the next stage, edge detection, will be demonstrated by considering the Gaussian filtered face.

4.2 Edge Detection

The ultimate goal of the observed structured light is to obtain height information, and to do this the lines must be distinguished from the other parts of the image. This is exactly the function of the edge detection processes if the difference between stripes and background is sufficiently large. An edge is an area of maximum rate of change in intensity. The simplest edge detector is any process which computes the derivative or gradients. Computing a derivative will yield the magnitude and direction of an edge and so vectors provide an ideal representation for calculation. If **f** is the edge in the image space then the first order derivative is,

$$\nabla \mathrm{f} = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} \qquad \text{Equation 4.8}$$

where G_x *is the horizontal gradient estimate*
G_y *is the vertical gradient estimate*

The magnitude (or intensity) of the edge is given by,

$$mag(\nabla \mathbf{f}) = [G_x^{\,2} + G_y^{\,2}]^{1/2} \qquad \text{Equation 4.9}$$

and finally the direction of the edge relative to the x axis is given by,

$$\alpha(x, y) = \tan^{-1}\left(\frac{G_y}{G_x}\right) \qquad \text{Equation 4.10}$$

However, derivative processes tend to react badly in the presence of noise and therefore in practice a smoothing element is useful. This has given rise to a number of edge detection schemes which are described here. Auto thresholding is done after edge detection to change the image to the binary form.

Thresholding is a relatively simple approach to segmenting an image into regions of similarity. The simplest method is the single band fixed threshold. This is based on first scaling the image so the pixels have values which lie between 0 and 1 (black and white),($0 \le v(i,j) \le 1$ where v is the value of the pixel at $[i, j]$).

Singleband fixed thresholding converts an image into binary form by application of the following process,

If ($v_{in}(i,j) >$ Threshold_level), then $v_{out}(i,j) = 1$
Else $v_{out}(i,j) = 0$

Binarisation of an image using a fixed thresholding technique depends critically on the value of the threshold that is used. A method of automating this process, computing the value of the threshold directly from the image, is used [70]. This technique for obtaining a suitable threshold level is obtain by searching for the two peaks in the edge

magnitude distribution and then searches for the suitable minimum value occurring between the two peaks. Also the image processing environment permits the user to apply a thresholding with hystersis on the image. This can be done by defining minimum and maximum grey values for thresholding.

4.2.1 Sobel Edge Detection

The Sobel operator consists of the two masks shown in equation 4.12 for computing the G_x and G_y components of equation 4.8. The magnitude component shown in equation 4.9 can be approximated with equation 4.11 to speed up the calculation.

$$\nabla f \approx |G_x| + |G_y| \qquad \text{Equation 4.11}$$

$$G_x = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \quad G_y = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \qquad \text{Equation 4.12}$$

The action of this set of masks is essentially that of a derivative process but with the desirable smoothing property which makes the Sobel operators superior in the presence of noise. The Sobel achieves this because the [-1 0 1] column/row is a differencing operator and the [1 2 1] column/row is a smoothing operator.

The result of applying Sobel edge detection following with thresholding is shown in Figure 4.9. Most of the lines have been detected but some, especially on the forehead and chin, have been missed and detected points merge. Also multiple responses to a single line have been produced because of the width of lines. This is a general problem for the first derivative operators (Sobel, Prewitt, Robinson and Kirsch [71]) when 2-D spatial gradient measurement is used along with a singleband fixed thresholding for binarisation the image. Therefore the Sobel operator is often used as a simple detector of horizontality and verticality of edges in which case only masks G_x and G_y are used.

(a)

(b)

Figure 4.9: Result of applying Sobel edge detection followed by thresholding on (a) median and (b) Gaussian

4.2.1 Laplacian Edge detection

Second order derivative edge detection techniques employ some form of spatial second order differentiation to accentuate edges. An edge is marked if a significant spatial change occurs in the second derivative. The edge Laplacian of an image function $f(x,y)$ in the continuous domain is defined as,

$$\nabla^2 f(x,y) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \qquad \text{Equation 4.13}$$

This is simply the second derivative of equation 4.9. The Laplacian operator is a very popular operator approximating the second derivative which gives the gradient magnitude only. The Laplacian is useful in that the equation 4.13 can be approximated for computational purposes as follows [72].

$$\nabla^2 f = 4z_5 - (z_2 + z_4 + z_6 + z_8) \qquad \text{Equation 4.14}$$

where z_N is an element in the convolution mask

This representation leads to the convolution mask as shown in equation 4.15.

$$h_1 = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \qquad \text{Equation 4.15}$$

If the Gaussian smoothing function is applied before using the Laplacian, then we have the Laplacian of Gaussian (LoG) operator. The result of LoG is shown in Figure 4.10 (a).

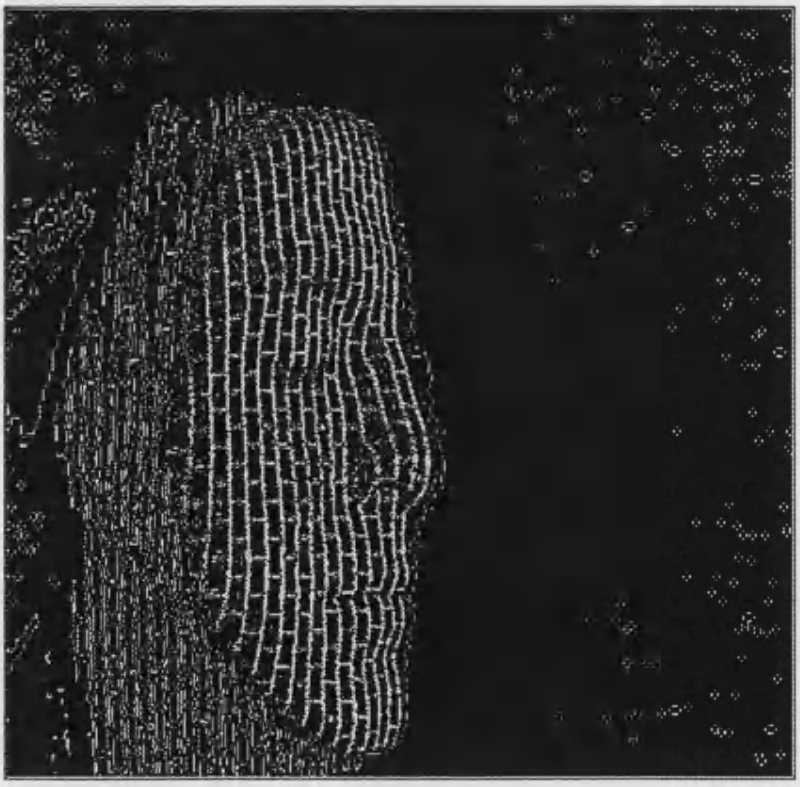
A Laplacian operator with stressed significance of the central pixel and its 8-neighbourhoods is sometimes used as shown in equation 4.16.

$$h_2 = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} \qquad \text{Equation 4.16}$$

The result of applying this mask is shown in Figure 4.10 (b)

The resolution of the lines with Laplacian edge detector is better than the Sobel edge detector especially because only one response is obtained for each line. Also the result of applying the mask h_2 is better than h_1 but still the image is full of the extraneous responses. The nature of the Laplacian edge detector stresses on the centre pixel and tends to form closed loop lines. Most of the extraneous responses on the sides of the image may be removed with a pruning function.

(a)



(b)

Figure 4.10: Result of applying Laplacian of Gaussian (LoG) operator on 'Face' with the mask (a) h_1, and (b) h_2.

4.2.3 Canny Edge Detector

The Canny edge detector [73] is designed to be an optimal edge detector for the application. The Canny operator works in a multi-stage process. First of all the image is smoothed by Gaussian convolution. Then a 2-D first derivative operator is applied to the smoothed image to highlight the edges. The algorithm then tracks along the detected edges to give a thin line in the output, a process known as non-maximum suppression. The tracking process exhibits hystersis controlled by two thresholds T_1 and T_2, with $T_1 > T_2$. The effect of the Canny operator is determined by three parameters, the width of the Gaussian kernel and the upper and lower thresholds [74].

The starting point is the operator based on the first derivative of the Gaussian function. Increasing the width of the Gaussian kernel reduces the detector's sensitivity to noise, at the expense of losing some of the finer detail in the image. The localisation error in the detailed edges also increases slightly as the Gaussian width is increased. Higher values for σ ($\sigma > 1$) pick out coarse detail whereas smaller values pick out finer details.

After the Gaussian is computed, the derivative will be calculated in four directions using the directional derivative masks shown in equation 4.17. These masks are applied in the positive and negative directions so that eight estimates of the edge magnitude and direction are obtained.

$$G_{xx} = \begin{bmatrix} 0 & 0 & 0 \\ -1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}, G_{yy} = \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}, G_{xy} = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, G_{yx} = \begin{bmatrix} 0 & 0 & -1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}$$

Equation 4.17 *Directional derivative masks*

When all the masks have been convolved for a given pixel, the maximum value of the gradient is found and that value is used as the edge value. To prevent several responses being produced for the same edge, a non-maximum suppression technique is used.

The non-maximum suppression scheme examines the magnitude of the gradients normal to the direction of the edge and if the original point is greatest it is taken as the edge. Figure 4.11 shows an example in which the edge at pixel **P(x,y)** is being estimated. The directional derivative masks have been applied and the edge direction and its normal have been calculated. The pixels nearest to the normal, **P(x+1,y+1)** and **P(x-1,y-1)**, then have their directional derivatives estimated and if **P(x,y)** is the greatest it is flagged as an edge.

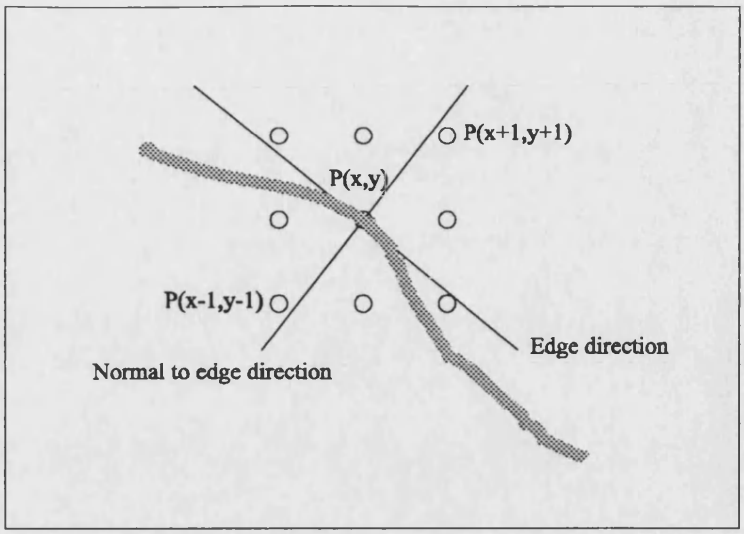


Figure 4.11: Non-maximum suppression of edge responses

It is possible to combine the smoothing and detection stages into a single convolution in one dimension, convolving with the first derivative of the Gaussian and looking for peaks.

The hystersis thresholder has been used in the Canny operator. An upper and lower edge value limits were defined for hystersis counters. Considering a line segment, if a value lies above the upper threshold limit it is immediately accepted. If the value lies below the low threshold it is immediately rejected. Points which lie between the two limits are accepted if they are connected to pixels which exhibit strong response. Setting the lower threshold too high will cause noisy edges to break up and setting the upper threshold too low increases the number of spurious and undesirable edge fragments appearing in the output.

Figure 4.12 shows the result of applying the Canny edge detector to image 'Face'. Note how the edge detection looks very detailed and how single width responses have been produced.

The advantages of the Canny operator are:
(a) Good detection: The ability to locate and mark all real edges.
(b) Good localisation: Minimal distance between the detected edge and real edge.
(c) Clear response: Only one response per edge.

The Canny edge detector achieves much better results than the Sobel and Laplacian edge detectors. The lines are much clearer and of approximately single pixel thickness which allows for better image analysis. The only problem with the Canny operator is that it may produce more than one pixel for a line in some specific areas of the image, e.g. at turning pixels on the lines and particularly on the junctions where three lines meet. This problem has been compensated for by applying the line following thinning algorithm on the Canny edge detected image, discussed later.

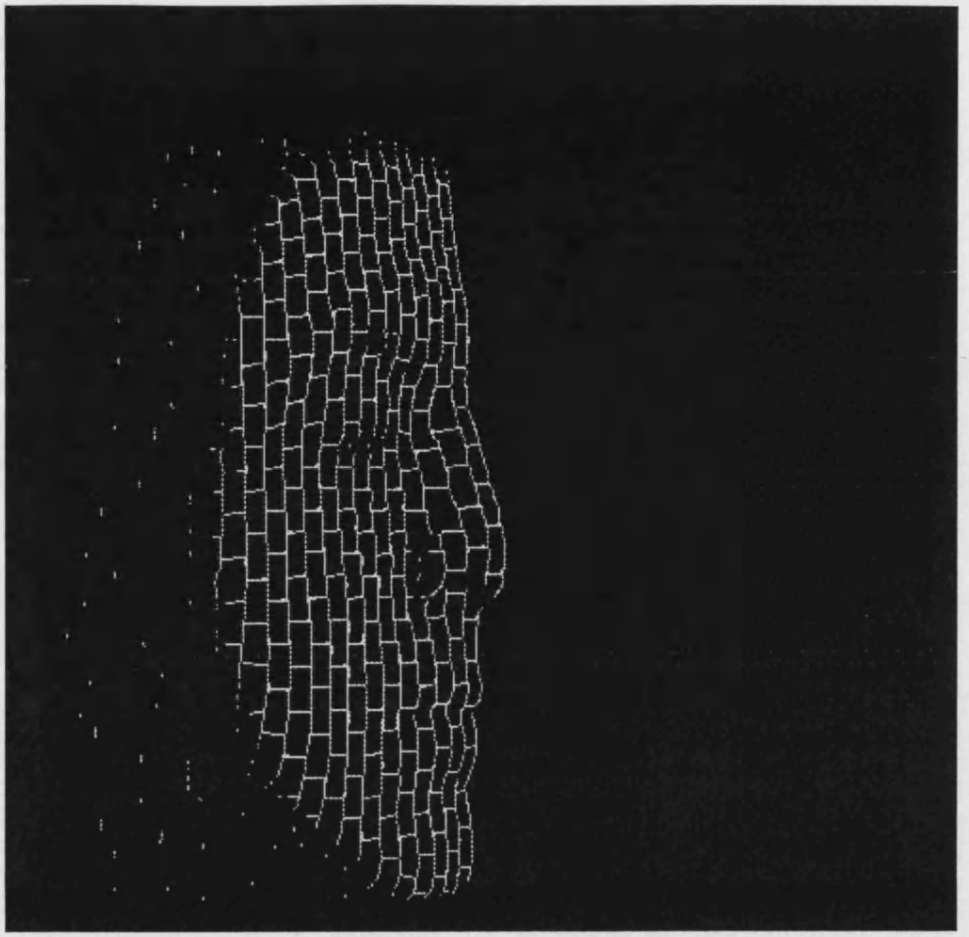


Figure 4.12: 'Face' after application of the Canny edge detector

4.3 Line Thinning

Thinning algorithms start with a binary image and by doing neighbourhood comparisons, erase outer pixels until a single pixel width skeleton structure remains [75]. The grid stripes are made exactly one pixel wide by shrinking each stripe to its skeleton. The skeleton of a region is obtained by stripping away boundary pixels in which removal would not break up a connected region into two separate pieces. The thinned lines obtained approximately trace the centre axis of the grid stripes as shown in Figure 4.13.

The Canny edge detector generates approximately a one pixel response, as shown in Figure 4.12. Because the Canny edge detector as implemented here uses non-maximum suppression, in the region of an edge only the strongest intensity pixel is flagged as a line. But a suitable thinning process is necessary to remove extra pixels on the lines which may occur, particularly in the region of junctions. Observation of the Sobel and Laplacian edge detected images, Figure 4.9 and Figure 4.10, indicate that they may well be improved by the application of line thinning. Therefore a thinning methodology has been applied on the edge detected images, to find the best skeleton and to show that if Sobel and Laplacian edge detected images can be made comparable to the Canny.

Available techniques for thinning could be classified into the three categories [76], (1) Sequential, (2) Parallel, and (3) Non iterative. In the sequential and parallel thinning algorithms, pixels are examined for detection based on the results of the previous iteration and which pixels are examined. The number of iterations depends on the thickness of the original figures in the picture. The non iterative methods based on producing a certain median or centre line of the pattern directly in one pass. These methods have the advantage of being computationally efficient. Because of the long processing time for iterative methods, and also possible discontinuities in the lines after each iteration, which would cause additional loss of information, they are not suitable for our application. A line thinning algorithm is considered an effective method when it (1) does not remove end points, (2) does not break connectedness, and (3) does not cause excessive erosion of the region.

A non-iterative method has been implemented based on windowing and line following for extracting the median pixels. The implemented method has been used for extracting the thinned response of the Canny and Laplacian edge detected images. Also the fast iterative thinning method [77] has been implemented to generate the thinned images from the Sobel and Laplacian edge detected images.

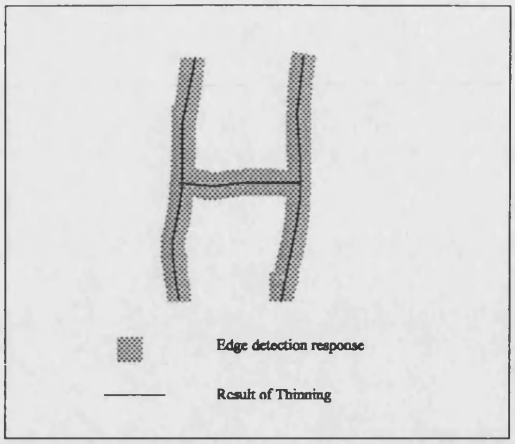


Figure 4.13: Thinning by finding the medial axes of the pattern

4.3.1 Line following technique

The simplest category of the non-iterative algorithms determines the midpoint by using a constant scanning direction [78]. It is valid in certain specialised applications where the scanning direction can be approximately perpendicular to that of the line. In other applications, the scanning directions are variable and may be selected by considering the direction of each line on the image [79].

Some algorithms obtain approximations of skeletons by connecting lines having certain orientations. For example, four pairs of window operations are used in four sub-cycles [80] to test for and determine the presence of vertical, horizontal, right or left diagonal parts in the pattern. At the same time, the operators also locate turning points and end points by a set of final point conditions, and these extracted points are connected to form a line segment approximation of the skeleton. In [81], the boundary pixels are first labelled according to the above four local orientations. For each boundary pixel, a search is made for the same kind of label on the opposite side of the boundary in the direction perpendicular to that given by the label. The mid-points of these pairs are then connected to form a skeleton.

Another set of algorithms determine the centre line P by tracking the two contours of each curve simultaneously. These trackers are located on both sides of a curve, and move under a common third criterion of minimum distance between them, the axis of the curve can be easily estimated as their average positions [82].

The main parts of the proposed algorithm are based on line following by using a flexible window [83]. The stages of processing are shown in Figure 4.14. The algorithm is performed by raster scanning the image. During the scan, the line follower is called to follow each encountered line. While following, the line follower erases the line and creates the skeleton instead. When the follower identifies a junction, it calls itself recursively to follow each of the branches.

The line following is based on two pointers and a dynamic window.

P_L: a pointer to follow the left edge of a line.
P_R: a pointer to follow the right one.
W: a rectangular window.

Initially the pointers are set by the scanner to the location of an encountered set pixel. To find this set pixel, the edge-detected image is scanned row by row and right to left, the algorithm reads each pixel value as it scans the row. When a white pixel is identified, the algorithm notes this pixel as the first pointer pixel $(P_R(x_R, y_j))$.

After finding the first pointer, the active window is applied to test for all the white pixel neighbours around this pointer. If the white pixels continue from the vertical direction, the line following is applied in the vertical direction and vice versa for the horizontal direction. The second pointer pixel $(P_L(x_L, y_j))$ is the left border in the window. The set pixel $(P_S(x_S, y_j))$ between P_R and P_L can be found by calculating the pixel density.

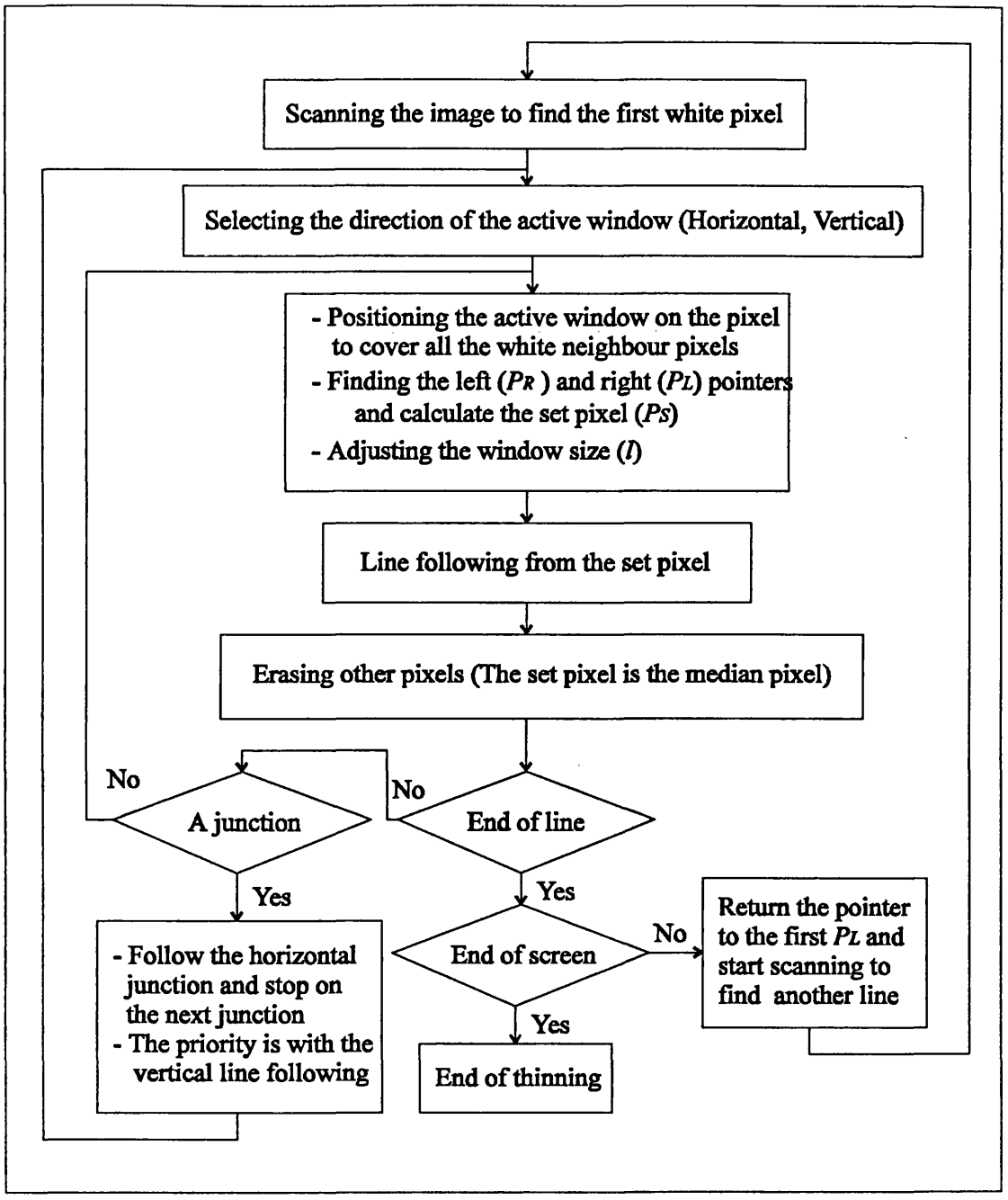


Figure 4.14: Line thinning procedure by scanning the image and line following

The line following is always done within the pre defined window W. The shape of the window has been defined so that it is easy distinguish between the line to be followed, and other segments of the image. This window is used in such a way that it surrounds the current location of the tracing pointers with centring of the set pixel (S). Each side of the window is set at a distance l to the P_L and P_R. The value of l is chosen that the window embraces the line and grows or shrinks according to the width of the line at the location of the window. It is also ensures that the centre of successive windows advances in the direction of the line following. The value of l influences the speed and accuracy of the algorithm. The bigger l means the bigger the window size which gives more accuracy but slow processing time.

After positioning the window on the set pixel $(P_S(x_S, y_j))$ and adjusting its size (l), the next row on the line and inside the window will be processed ($j+1$). This is done by searching inside the window and finding the right and left pointers again and then calculating a new set pixel for the processing row $(P_S(x_S, y_{j+1}))$. A new window will be generated on the new set pixel to continue the process as before. By generating the window, the white pixels around the set pixel will be erased to complete the thinning process. The set pixels have been used to show the median pixels of a line.

The value of l is set to 2 to give the necessary accuracy with some degree of smoothing. If the number of the pixels on the next row ($j+1$) are more than the window size then the thinning algorithm will consider only the pixels which are surrounded by the window. This is important to have a smooth skeleton.

The line following has been applied in two directions, vertical and horizontal, but the vertical line following is the main part of the thinning process. If a junction is found during the vertical line following, the horizontal line will be processed until meeting another junction. At this situation the algorithm stops the horizontal following and the pointers will go back to the original junction point and the vertical following will be continued. The processes of vertical and horizontal line following are shown in Figure 4.15 and Figure 4.16.

During the line thinning, all the pixels are removed from the screen and only the median pixels are stored in the thinning array as a skeleton. Removing the processed pixels from the image is necessary to find other lines on the image.

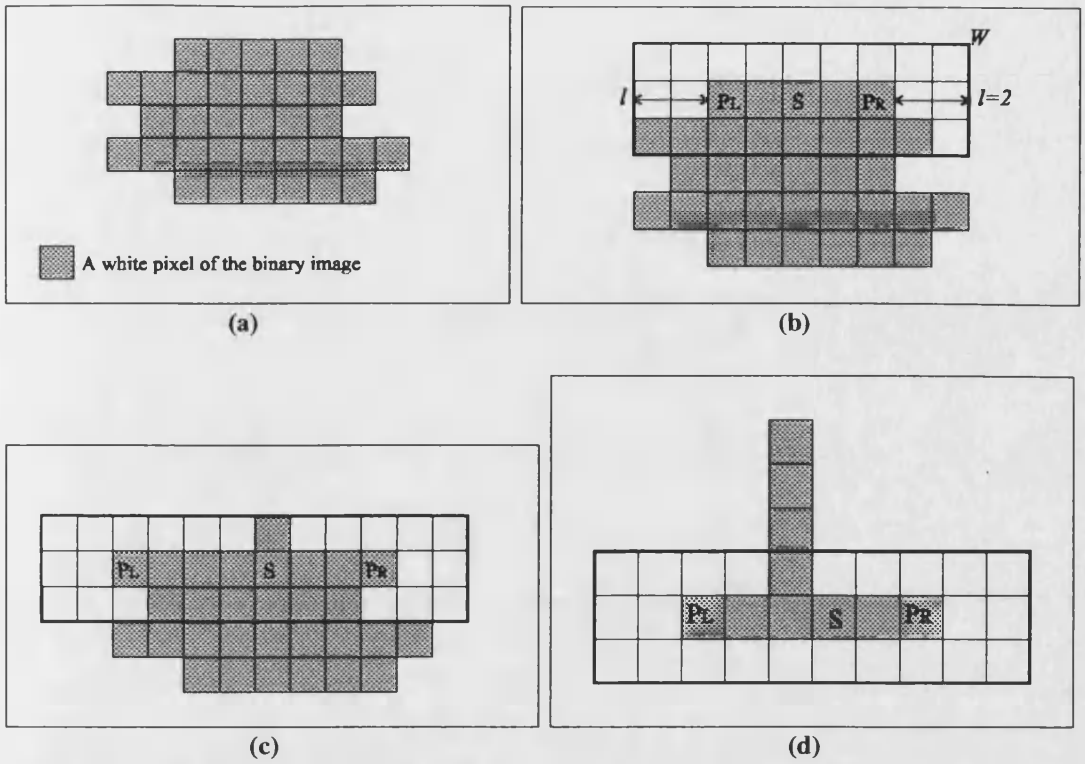


Figure 4.15: Line thinning by using the vertical line following strategy. (a) A binary line, (b) applying the window and positioning the pointers, (c) extracting the median pixel and moving the window to the next row (d) End_of_line situation and the resultant skeleton

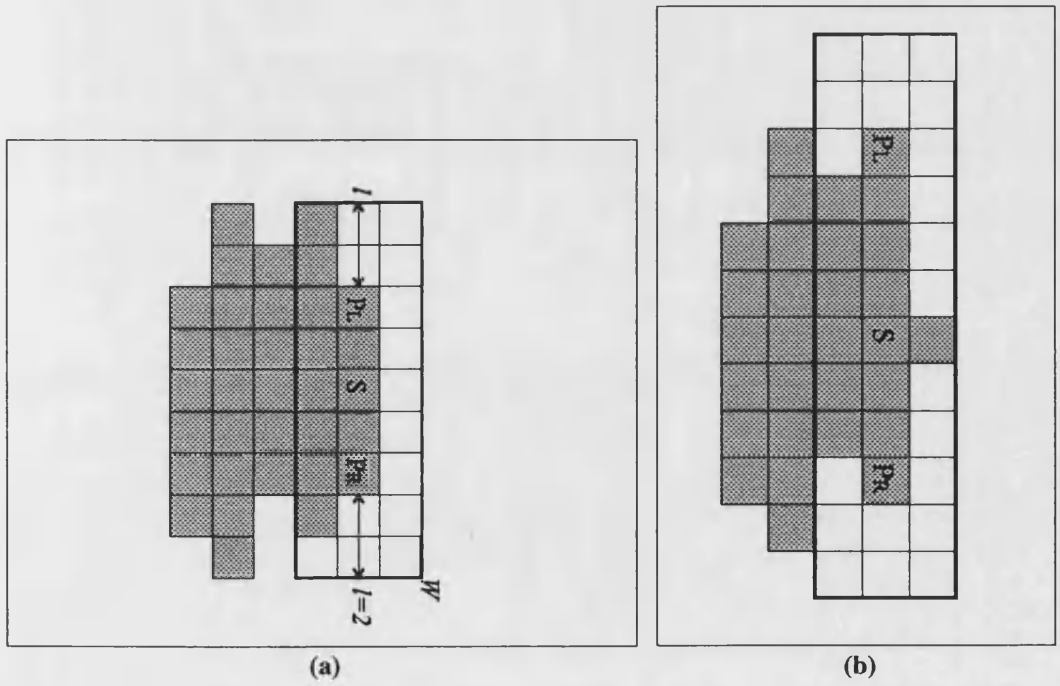


Figure 4.16: Horizontal line following, (a) positioning the window, and (b) growing the window to thin the line.

The pixel following process can be divided into four cases, depending on the relationship between the current window and the current line segment:

- In the direction of following, the size of the current window covers the line and thus both boundaries of the line are inside of the window. This is the most common case that occurs during line thinning.

- Junction detection: During the vertical line following a junction can change the direction of the thinning to horizontal line following. This process is continued until meeting a new junction. When this happens the horizontal following is stopped and the pointers go back to the first original junction and the vertical line following will be applied on the line.

- End_of_line case: In the direction of the following, the line ends inside the current window and thus it is considered as end_of_line as shown in Figure 4.15 (d). As demonstrated in Figure 4.17, when the end_of_line is detected by the program, the pointer returns to the first left pointer on the thinned line and then the horizontal scan is started again to search for another unprocessed line on the screen. Because of thinning the processed line, the horizontal scan can distinguish between the processed or un-processed lines. The line following and scanning the screen is continued until all lines on the screen have been processed.

- During the thinning, black pixels, caused by noise on the actual surface, can occur within the unthinned rows of a line. When this happens the algorithm continues to follow assuming all the pixels are white. If the black pixels occur for more than five consecutive rows a branch in the line is assumed and each branch is processed separately.

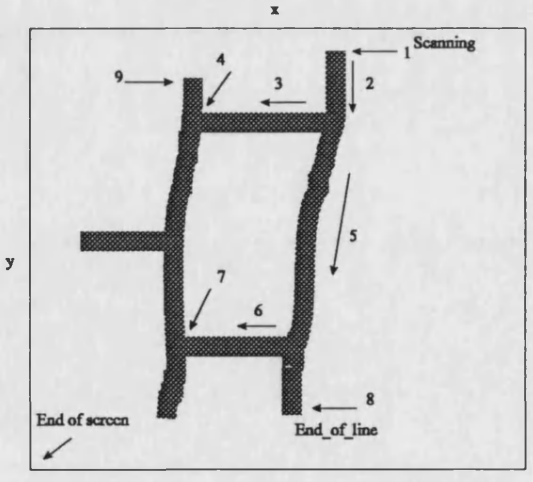


Figure 4.17: Detection of a junction, End_of_line and continue scanning to find another line, 1) horizontal scanning to find the first pointer on the first line, 2) vertical line following to thin the line, 3) detection of a junction and start horizontal line following, 4) find another junction and stop the line thinning, 5) back to the previous junction point and start vertical line following again, 6) find another junction, stop the vertical following and start the horizontal following again, 7) junction detected, stop the horizontal following and go back to the previous junction point and continue vertical following, 8) End_of_line detection, 9) Return the pointer to the first left pointer on the line to find the next line for processing, Scanning again to find another line. Detection of a new line, continue the process until end of screen.

The implemented thinning method is suitable for applications when the lines and junctions are clear on the edge detected image. In images, that result from Sobel edge detection, the closed contours prevent the application of line following and so an iterative technique for line thinning has been implemented.

4.3.2 Iterative Technique

To compare the results of the implemented thinning method with other available thinning techniques, a fast parallel method has been chosen. The iterative method consists of successive passes of two basic steps applied to the contour points of the given region, where a contour point is any pixel with value 1 and having at least one 8-neighbour value 0. With reference to the 8-neighbourhood definition shown in Figure 4.18, step 1 flags a contour point p for deletion if the following conditions are satisfied:

p_9	p_2	p_3
p_8	p_1	p_4
p_7	p_6	p_5

Figure 4.18: Neighbourhood arrangement used by the thinning algorithm

(a) $2 \leq N(p_1) \leq 6$

(b) $S(p_1) = 1$

(c) $p_2 \cdot p_4 \cdot p_6 = 0$ Equation 4.18

(d) $p_4 \cdot p_6 \cdot p_8 = 0$

where $N(p_1)$ is the number of nonzero neighbours of p_1; that is

$$N(p_1) = p_2 + p_3 + \ldots + p_8 + p_9 \quad \text{Equation 4.19}$$

In step 2, conditions (a) and (b) remain the same, but conditions (c) and (d) are changed to

(c`) $p_2 \cdot p_4 \cdot p_8 = 0$

(d`) $p_2 \cdot p_6 \cdot p_8 = 0$ Equation 4.20

Step 1 is applied to every border pixel in the binary region under consideration. If one or more of conditions (a)-(d) are violated, the value of the point in question is not changed. If all conditions are satisfied the point is flagged for deletion. After step 1 has been applied to all border points, step 2 is applied to the resulting data in exactly the same manner as before. Thus one iteration of thinning algorithm consists of (1) applying step 1 to flag border points for deletion; (2) deleting the flagged points; (3) applying step 2 to flag the remaining border points for deletion; (4) deleting the flagged

points. This procedure is applied iteratively until no further points are deleted, at which time the algorithm terminates, yielding the skeleton of the region.

4.4 Pruning

Pruning methods are an essential complement of the thinning and skeletonizing algorithms, because these procedures tend to leave parasitic components that need to be cleaned up by post processing [72]. The problems in the images obtained were investigated and then the pruning algorithm developed to solve them.

The pruning algorithm wants to remove noisy pixels from the resulting skeleton as much as possible by applying four different windows as shown in Figure 4.19 . The image is scanned from right to left and each white pixel (*S*) will be examined by its neighbours. By considering the condition of neighbourhood pixels in each window, the algorithm decides to keep or remove the pixel (*S*) and also the pixels shown with star (*). If all the neighbourhood pixels according the windows are black then the white pixel (*S*) and don't care pixels (*) will be changed to black. If not, the defined window will be applied for the next pixel followed by the scanning direction and the test is done. The pruning window stops at the end of each screen and a new one starts again at the first co-ordinate of the screen.

$$\left\{\begin{bmatrix} p_1 & p_8 & p_7 \\ p_2 & S & p_6 \\ p_3 & p_4 & p_5 \end{bmatrix}, \begin{bmatrix} p_1 & p_{10} & p_9 \\ p_2 & S & p_8 \\ p_3 & * & p_7 \\ p_4 & p_5 & p_6 \end{bmatrix}, \begin{bmatrix} p_1 & p_{12} & p_{11} & p_{10} \\ p_2 & * & S & p_9 \\ p_3 & * & * & p_8 \\ p_4 & p_5 & p_6 & p_7 \end{bmatrix}, \begin{bmatrix} p_1 & p_{12} & p_{11} \\ p_2 & S & p_{10} \\ p_3 & * & p_9 \\ p_4 & * & p_8 \\ p_5 & p_6 & p_7 \end{bmatrix}\right\}$$

Figure 4.19: The pruning windows for removing noisy pixels. If all of the neighbourhood pixels of the set pixel (p_1, p_2, ..., p_n) are black, then erase (S) and (*), i.e. change to black. If not, then follow the process for the next pixel.

The noisy pixels around the face from the Laplacian edge detection or other edge detections have been removed by applying a defined mask for cleaning them from outside the face. The pruning of the image boundary and also lines and junctions inside the image is a process which will be done during the tracing algorithms, explained in chapter 5.

The results of the thinning and pruning algorithms on the edge detected images are shown in Figure 4.20 to Figure 4.23. Most of the noisy pixels around and inside the processed faces are removed. Figure 4.20 shows the thinned images of the Sobel edge detection for both median and Gaussian enhancement obtained by applying iterative thinning and pruning. For each projected line, two thinned line are available (transition of the black to white and vice versa). The resultant image from Sobel on the median is similar to the Sobel on the Gaussian are more evident.

The results of the line following thinning algorithm and pruning on the Laplacian of Gaussian (LoG) are shown in Figure 4.21. The results of applying the iterative thinning

on the edge detected images (LoG) followed by pruning are shown in Figure 4.22. Although the noisy pixels around the face are removed by applying the pruning masks, some of the pixels on the lines have also been removed because of the noisy structure of the Laplacian edge detection. The results of the iterative thinning on LoG are better than the line following, because this gives better localisation of thinning at each iteration for the image.

The Canny edge detected face after the line following thinning and pruning is shown in Figure 4.23. Because iterative techniques are well known for producing discontinuities on thinned lines, only the line following technique is used on the Canny edge detected image. The resultant image is one pixels thickness and suitable for the post processing.

(a)

(b)

Figure 4.20: Face after applying the iterative thinning and pruning on (a) Sobel of median and (b) Sobel of Gaussian.

(a)

(b)

Figure 4.21: Line thinned images of the face (LoG) by using the line following method and pruning, process on Figure 4.10.

(a)

(b)

Figure 4.22: The iterative thinning and pruning on LoG, process on Figure 4.10

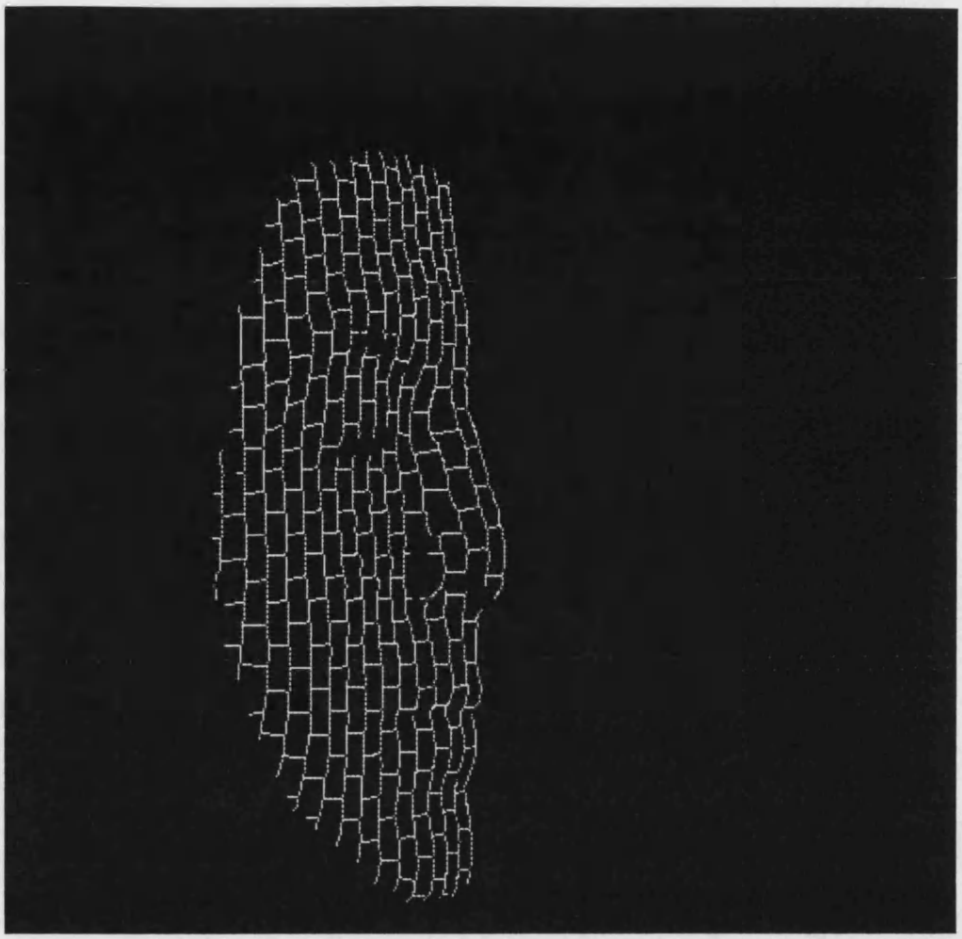


Figure 4.23: Canny edge detected 'face' after applying the line following thinning and pruning

4.5 Conclusion

The image processing software that has been written for the auto-extraction stage includes enhancement, edge detection, thinning and pruning. Minimum hand intervention and processing time were considered when writing the algorithms to achieve a real time and automatic algorithms.

The overall contrast of the images are already good due to adjusting the AGC manually in the frame-grabber from which the image was digitised. Contrast around the lines and specially on the cheeks, however, could be improved and this is why histogram specification was used. Adaptive histogram equalisation was tried but although some contrast enhancement was achieved, there was also an increase in noise in low image content areas. This noise produced faulty edges when the edge detection process was applied and therefore adaptive contrast enhancement was not used in conjunction with other techniques. It could still be applied, but with a much more intelligent algorithm which enhanced only in the defined area.

The pre-processing steps for image processing are the median filter and the Gaussian filter, the results of which are shown in Figure 4.7 and Figure 4.8 respectively. The median filter has removed most of the noise from the image. The Gaussian filtered result is very similar with the median filter result. The Gaussian filter is also an integral part of many edge detection processes and so we shall favour its use over the median filter. But to find which one is the most suitable at minimising the noise, the results of the face after pre-processing, edge detection and thinning will be compared later.

The enhanced image is then changed to an edge detected image by a suitable edge detector. The result of the Sobel edge detector is shown in Figure 4.9. Here most of the lines have been detected but some, especially on the forehead and chin, have been missed and detected points merge. This is due to the resolution of Sobel operator which needs more pixels to work with. One very important observation of the Sobel edge detected image is that due to the width of lines, multiple responses to a single line have been produced. The resolution of the lines with the Laplacian edge detector is much better than the Sobel operators as shown in Figure 4.10, but the image is full of the extraneous responses. The nature of the Laplacian edge detector stresses on the centre pixel and tends to form a closed loop lines. The Canny edge detector solves all the problems, as shown in Figure 4.12. Because the Canny edge detector as implemented here uses non-maximum suppression, in the region of an edge only the strongest intensity pixel is flagged as an edge. This is why the Canny image has much less marked points than the Sobel and Laplacian images.

Selecting a suitable thresholding level is one of the most important parameter to have a clear binary image. An auto thresholding method is used after edge detection to find the lower and higher gray levels on the lines, and then calculating the threshold level based on them. Also selecting the lower and higher hystersis values for the Canny edge detector have been done for achieving a high resolution image.

A property of all edge detectors when applied to the image is that they produce an edge response for the light-dark transition as well as for the dark-light transition. This is covers the area of several pixels when encoded onto the image array. This is

contrary to thinning methodology, which produce a single response that should be contained within the midline of the lines. The line following thinning algorithm produced a one pixel wide skeleton of the edge detected image by generating the median of the grid stripes. The idea for implementing the line following thinning algorithm was to find the line skeleton especially to clean the Canny edge detected image therefore this algorithm is not suitable for a closed contours such as happened in the Sobel edge detection. To compare the results of thinning on Laplacian and also to produce the thinned image for the Sobel edge detector, a fast iterative technique was used.

By comparing the results, it is clear that the 'face' after applying the Canny edge detection following by the line following thinning and pruning is the most appropriate image for the interpretation stage. A description of the higher level image processing is given is chapter 5 by introducing the tracing algorithms and then the traced image will be ready for reconstruction.

Chapter 5

Feature Interpretation

Following the image processing steps, we now have a binary image consisting of single pixel width pattern which show where the projected lines are. The contours of the human face mean that certain parts of the projected grid can not be seen on the captured image (dead shadow areas) and breaks occur in the projected grid itself around the eyes, eyebrows and nose. In order to produce a 3-D data set, all of the lines must be perfectly formed, of single-pixel thickness and be continuous. In this section, the grid lines are interpreted and decisions must be made as to whether any given pixel belongs to which line. The algorithms in this section attempt to provide solutions for both noise and breaks and to fill the disconnected areas.

The main stage involved in this procedure is tracing. A novel approach of tracing includes line and loop tracing, introduced to overcome the problems encountered when only vertical lines were used. The first stage of assigning heights is to label the grid junctions and then use the calibration matrices to calculate the actual height. The two dimensional image after labelling is transformed to the 3-D information and will be used for the reconstruction stage.

The final part of this chapter looks at how the 3-D data set may be displayed as a helpful representation for medical applications. A user friendly graphics environment has been developed to produce either a wire frame or rendered model of a face, and by using the mouse facility this model can be manipulated in three dimensions. Also the joining algorithm, by using the feature matching technique, will be used to reconstruct a complete representation of a face.

5.1 Introduction

As stated earlier, the correspondence problem in stereo matching is to identify feature-pairs in the two images that correspond to the same physical surface feature. Using structured light, the feature points are those pixels made bright by the structured light pattern. If a single light beam creates a single bright dot in the image or a single light plane creates a single bright stripe curve in the image, the correspondence problem is immediately solved, the dot or the stripe curve is the only matching candidate [84]. An entire grid of lines can be projected on the scene at once. Suppose a light stripe is seen by the camera, the matching procedure must find out which line on the slide is the line that generated the stripe. Thus, the correspondence problem becomes the line labelling problem. If each grid line has a unique property; colour, thickness, code, etc., identifying grid lines is no more difficult than in the single stripe case, since the unique line is the only candidate.

Therefore a key step in the 3-D reconstruction using structured light is to solve the grid labelling or matching problem, to find the correct correspondences between the detected grid points in the camera image and the points in the projected grid pattern. The goal is to identify all the line segments in the image by comparing their pixels against those in the projected sequence. The problem is important since the lines will typically be broken due to surface discontinuities and image processing failures.

Some of the engineering approaches that have been tried to solve the correspondence problem [85] include time modulation of the projected grid pattern, colour coding of the grid strips and spatial labelling. For a given point on the pattern, the corresponding point on the image must lie on a line whose equation can be derived from the geometry of the system. These lines are called 'epipolar' lines. Their benefit derives from the fact that they reduce a two-dimensional search for possible matching points into a one dimensional problem. Some of the basic methods for matching are explained here.

The time modulation technique implemented by *Posdamer* and *Altschuler* [86], is based on the use of a dot matrix of n×n binary light beams. Each column of the pattern can be independently controlled, so it can be turned on or off. Then, several masks can be made as a sequential projection of different patterns. The number of patterns to be projected is determined by the number of columns to be coded. The system usually works as follows. Firstly, a whole illuminated dot pattern is projected on the scene. The camera images all the dots reflected from the surfaces of the scene and keeps their positions in a datum structure. In the following steps, each coded pattern is projected and the system can ask for the information in the stored positions. After all the patterns have been projected, each detected point has a code that identifies the beam column to which it be lays.

Boyer and *Kak* [87] implemented a colour coding scheme that projects a single pattern of red, green, blue and white vertical stripes. Optionally, between each slit, a black gap can be placed. This gap can be used as a slit separator. In this case, the correspondence problem is limited to obtaining the relationship between the imaged slit and the projected one.

Vuylsteke and *Oosterlinck* [88] developed a spatially coded structured light system that can uniquely label grid points from a single-camera image. The scene is illuminated by a chessboard pattern of light and dark squares. The vertices of these squares are each marked by a single bit in the form of a light or dark spot. A novel application of error-correcting binary codes determines the column index of each grid point from the local pattern of bit marks.

LeMoigne and *Waxman* [29] investigated the feasibility of a structured light range finder for autonomous robots, and studied a number of operational considerations, including the use of a spatially marked grid. A grid of vertical and horizontal stripes was used with a vertical baseline so that the detected vertical stripes were epipolar lines in the camera image, and the vertical stripes could be easily detected and their identity determined. Several illuminated squares ('dots') were included in the projected grid, and were used to guide the grid labelling. Labels were first assigned to intersections along vertical stripes bounding the detected dots and the identity of the horizontal stripes was determined by tracking along the horizontal stripes from these intersections. A relaxation-based procedure [89] was used to fill in missing grid points.

Stockman and *Hu* [90] have studied grid labelling as a constraint satisfication problem. Constraints on possible labels assigned to individual grid points and to pairs of neighbouring grid points are propagated using a discrete relaxation procedure to find a globally consistent set of grid labels that satisfy all of the constraints. They classified the labelling constraints into geometric and topological constraints. Geometric constraints reflect knowledge of the stereo geometry and include the epipolar constraint, unique imaging of a single projected ray and the requirement that illuminated object points along the same stripe lie in a plane. Topological constraints reflect the assumption that a connected or smooth stripe in the camera image is probably the image of a connected or smooth grid stripe on the object's surface. An initial list of possible labels is assigned to each grid point in the camera image using the epipolar constraint. The constraints are then propagated to find a consistent labelling.

5.2 Feature tracking

As a pre-processing required before matching, the pixels detected during the feature extraction process are grouped into continuous lines using the tracing algorithms. These tracing algorithms exploit the fact that we know that the projected pattern was continuous, and allow us to recover the structured light image.

Researchers have developed many techniques to "code" the light stripes in multi-pattern situations [91 , 92] so that triangulation can be applied, but few have studied the use of structured light with only one pattern for extraction of surface characteristics. The objective of designing the new algorithms were to allow matching in scenes that contain deformities, discontinuities and overlapping. The emphasis has been on using all the ordered information in a structured light image by applying only one pattern to assist automatic tracing and matching and therefore moves away from the raster-scan approach.

A spatial marking scheme has been used that fixes T-junctions along the vertical parallel lines by considering the reference line (line 0) on the profile of the face as shown in Figure 5.1. The structured light system uses two cameras and a projector with horizontal baseline set-up so that the epipolar lines are horizontal in the grid slide plane of the projector. The horizontal lines are distributed between the vertical lines to make a brick pattern. The distance and position of the vertical and horizontal lines on the grid is known. If d_g is the distance between each vertical line on the slide, the distance between each horizontal line is $2d_g$. Each grid junction is labelled to show its position on the slide.

Since each T-junction of the grid stripes is set at the middle of the two neighbour T-junctions, a grid point is uniquely determined from its position on the slide co-ordinate system. The advantage of using a brick pattern instead of a normal grid with only cross junctions of parallel vertical and horizontal lines is having more junctions, six instead of four, for each defined square inside the pattern.

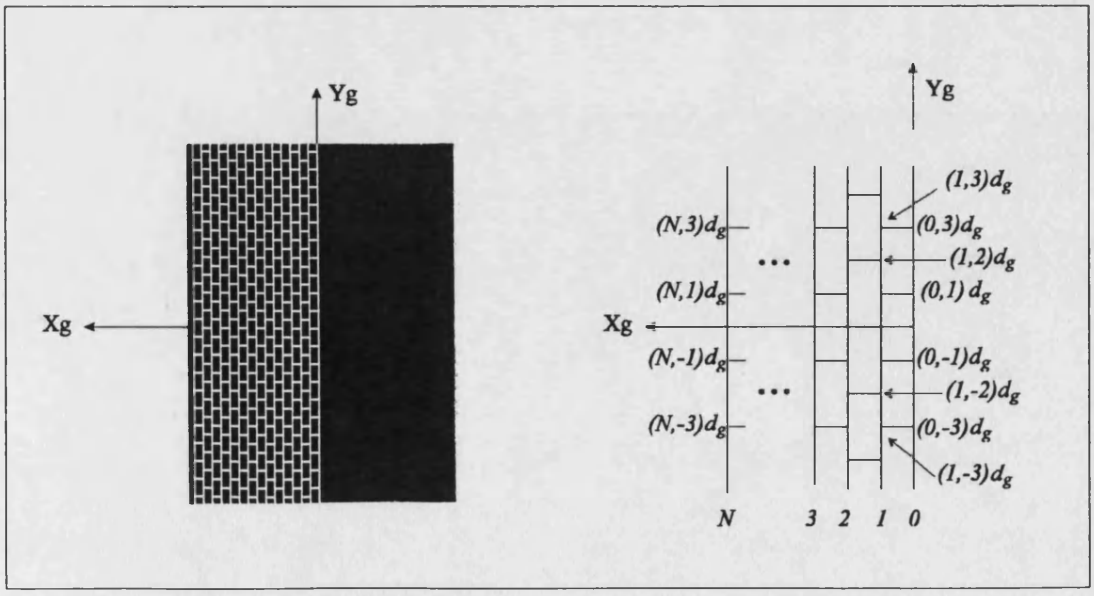


Figure 5.1: Brick pattern for the left side of a face and the slide co-ordinates

The stages involved for implementing the labelling are shown in Figure 5.2. The novel approaches were presented to correct the discontinuities in the image by using the tracing algorithms and then the labelling algorithm will be applied to ready the image for the reconstruction stage.

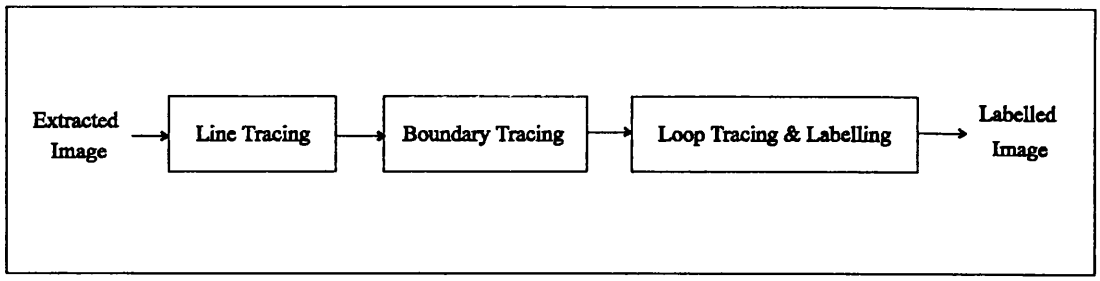


Figure 5.2: Matching stages

The work starts by preparing the profile line as the reference line by applying the line tracing algorithm which defines the priority areas to scan the correct pixels and the filling strategy to seal the discontinuities on the line. The main stage for recovering the structured light image is the loop tracing algorithm. The image after applying the brick pattern can be divided to a series of small loops, contours surrounded by four corner junctions, that are distributed between two vertical lines. The loop tracing algorithm scans inside each loop and then completes any discontinuities by using the junction orders. The difference coding sequence and junction sequence are novel methods for extracting and classifying the problems inside the loops, to be described in section 5.1.5. The loops positioned on the boundary of a face are exceptional from other loops and must be labelled before starting the loop tracing by using the boundary tracing algorithm. This finds all the boundary information of the image such as the boundary junctions and end points. The information is saved in the boundary network and will be used during loop tracing. A perfect reference line from the one profile side and the boundary network from the remainder of the image permit the loop tracing to process the loops. When all the loops are perfectly formed, the labelling algorithm is applied to index the junctions and vertical lines. Accurate labelling using the marked grid can be accomplished using the information obtained during the line and loop tracing stages. This makes the labelling easy and robust in comparison with other implemented techniques based on propagating the defined constraints using the relaxation technique such as proposed by *Stockman* and *Hu.* [93].

To write an effective algorithm for covering all types of deformities on the face, all the foreseeable problems for lines have to be considered and then compensated for. After processing many images, the common difficulties are:

- Small discontinuities, less than five pixels, on the line caused by the low contrast (or concave area), on some parts of a face such as eyebrows, eyelashes, eyes, nasal cavity, lip lines and any type of hair on the face.

- Large discontinuities, greater than five pixels, on a line caused by missing the projected line on a sharp edge or in a hole. This problem usually happens on the edge of nose, around and inside the ear and the connection point of the neck with face.

- Overlapping of the vertical lines because of any deformity and sharp area on the face. With this condition two lines cross each other and therefore overlapped pixels will be produced. This problem usually happens near the curvature of a nose.

5.2.1 Reference Line Tracing

The profile line as a reference line is the best place for starting the process so it must be perfectly formed by applying the line tracing algorithm. The profile line has been used for labelling and correcting the other vertical lines. Figure 5.3 illustrates the feature extracted image of the 'face'.

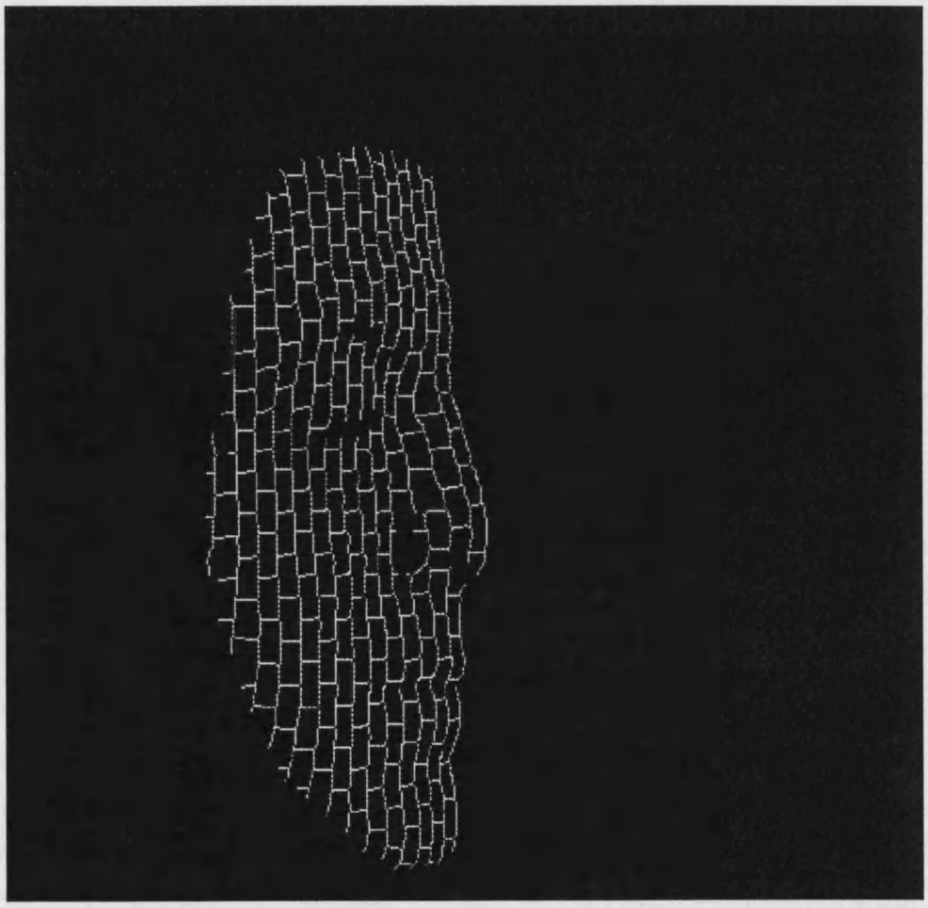


Figure 5.3: Extracted image of the 'face'

As we know that pixels on the right side of the reference line must be black so the scanning procedure can follow the pixels by looking to the right side and selecting the most suitable direction. The start point for tracing is found by overlapping a vertical line with the reference line as shown in Figure 5.4. Two tracing directions are defined to scan the line from the co-ordinates of the overlapping point; forward and backward line scanning. If the co-ordinates of the overlapping point are considered as (x_o, y_o) then each scanning direction is;

- Backward scanning on the reference line for the pixels with $y < y_o$.
- Forward scanning on the reference line for the pixels with $y > y_o$.

If the vertical line crosses the profile line on two different pixels instead of one, the trace will be done from each of the pixels separately and the filling algorithm will correct the discontinuity area between the disconnected points.

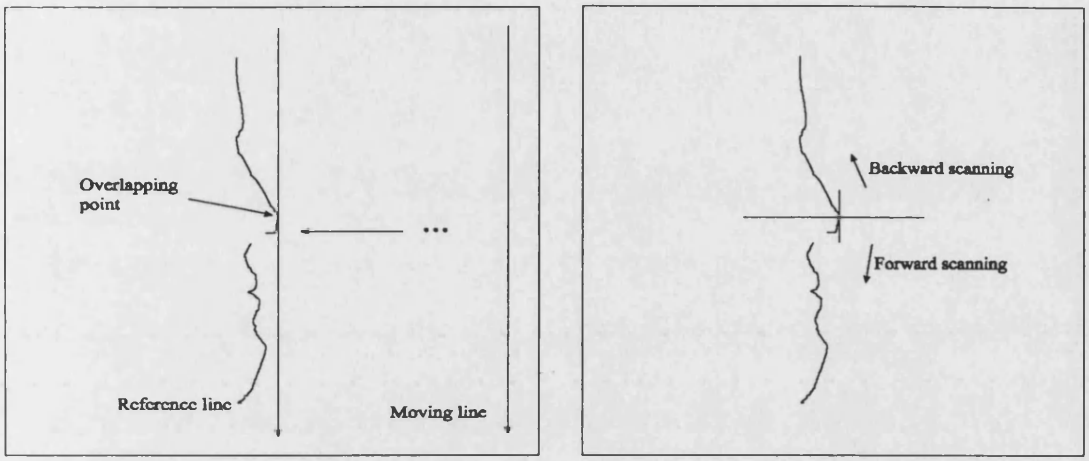


Figure 5.4: (a) The overlapping pixel of the reference line (profile line) and the moving vertical line, this pixel is the start point for tracing on the reference line. (b) Forward and backward scanning from the overlapped point on the reference line

A difficult problem of the tracing procedure is to determine the scanning direction. At each line point, its eight adjacent pixels are the candidates for the next element. If there is a single pixel in this neighbourhood, the selection of direction in which to continue is obvious. However, there are often two or more pixels present in the neighbouring eight. Also in some cases the neighbourhood pixel is not positioned in the adjacent eight and the algorithm has to search for its position. Several strategies have been introduced for tracing the gray or binary images [94 , 95], such as edge intensity, edge magnitude, edge direction, line direction and local tracing shape prediction. The implemented line tracing algorithm has a neighbourhood window type structure with different priority values for each neighbour.

The algorithm has been implemented to find the next pixel of the scanning process by defining scan areas and then applying a suitable scanning direction inside that area. First consider that the next pixel is one of the eight adjacent pixels. The question is how can we index the next pixel among the neighbourhood pixels or what is the priority for each neighbouring pixel as a next pixel. Figure 5.5 shows the candidate pixels for the next pixel along the adjacent pixels for the forward and backward scanning. If a direction is assigned for each adjacent pixel by considering the main pixel, the priority number for each direction will be found as follows;

- For the backward scanning, the right side of the main pixel is a black pixel until achieving the next pixel
- For the forward scanning, the left side of the main pixel is a black pixel until achieving the next pixel.

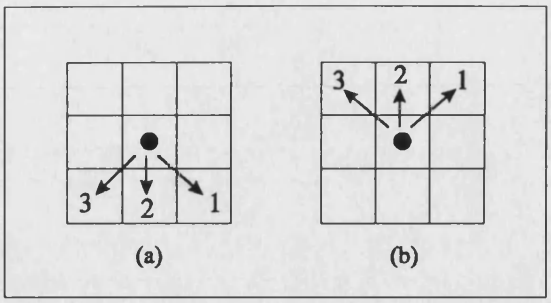


Figure 5.5: The candidate neighbourhood pixels as a next pixel and the directional priority for, (a) forward scanning and (b) backward scanning.

Figure 5.5 shows that if a main pixel has two neighbouring pixels in the directions of 1 and 3, the next pixel will be the pixel with the higher priority, i.e. direction 1. The extension of the directional priority is an area searching when the next pixel is not found around the neighbouring area. The area searching for the next pixel must be adapted with the above definitions. Figure 5.6 shows how the search should be applied on the image and what is the priority direction for pixels on the search areas.

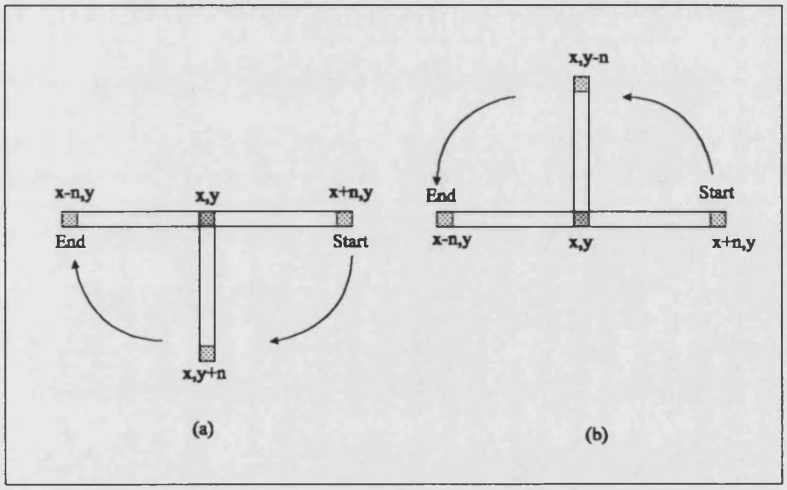


Figure 5.6: Priority directions for (a) forward, and (b) backward scanning

The algorithm has been implemented to find the next pixel of the scanning process by defining scan areas and then applying a suitable scanning direction inside that area. The scanning area is divided into the three sub-regions for each scanning direction as shown in Figure 5.7 and Figure 5.8. The first scan area is chosen to find the neighbouring pixels with the greatest probability. If the next pixel is not found in the first scanning area, the second scan area will be used with directional searching and priority values for each pixel given in Figure 5.7. Ultimately, the next pixel will be searched for in the scan area three. Each scan area covers 15 by 15 neighbourhood pixels around the current pixel. The scanning areas and their element priorities have been chosen to satisfy the conditions for the next pixel on the reference line.

	x-1	x	x+1
y	*	S	*
y+1	3	2	1

(a)

	x	x+1	x+2	...	x+14	x+15
y	S	*	*	...	*	*
y+1	*	*	14	...	2	1
y+2	30	29	28	...	16	15
:	:	:	:	...	:	:
y+15	238	237	236	...	224	223

(b)

x-15	x-14	...	x-2	x-1	x	
*	*	...	*	*	S	y
210	195	...	15	*	*	y+1
211	196	...	16	1	*	y+2
:	:	:	:	:	:	:
224	209	...	29	14	*	y+15

(c)

Figure 5.7: The sub-scanning areas for the forward scanning, the current pixel is the set pixel (S) and the priority for the next pixel is always 1, 2 and etc. (a) the neighbouring elements, (b) the second scan area by searching for the next pixel from left to right, (c) the third scan area by searching for the next pixel from top to bottom.

	x-1	x	x+1
y-1	3	2	1
y	*	S	*

(a)

	x	x+1	x+2	...	x+14	x+15
y-15	238	237	236	...	224	223
:	:	:		...	:	:
y-2	30	29	28	...	16	15
y-1	*	*	14	...	2	1
y	S	*	*	...	*	*

(b)

x-15	x-14	...	x-2	x-1	x	
224	209	...	29	14	*	y-15
:	:	:	:	:	*	...
211	196		16	1	*	y-2
210	195	...	15	*	*	y-1
*	*	...	*	*	S	y

(c)

Figure 5.8: The sub-scanning areas for the backward scanning

The star pixels (*) in the defined areas have been previously observed or are not admitted pixels. The size of the scan area was determined by processing many captured images and studying the range of discontinuities on their reference lines. In general because of the accurate patient positioning for capturing the image and aligning the first line, the reference line usually has only a limited number of discontinuities primarily on the sharp areas such as nose and lip.

When the next pixel is found any gap is filled by the line filling algorithm.

5.2.2 Line Filling

The purpose of filling is to add pixels onto a line, vertical or horizontal, to stop any gaps and this action is implemented with a program which runs through the traced line for sealing the discontinuities. A perfect line and a disconnected line are shown in Figure 5.9.

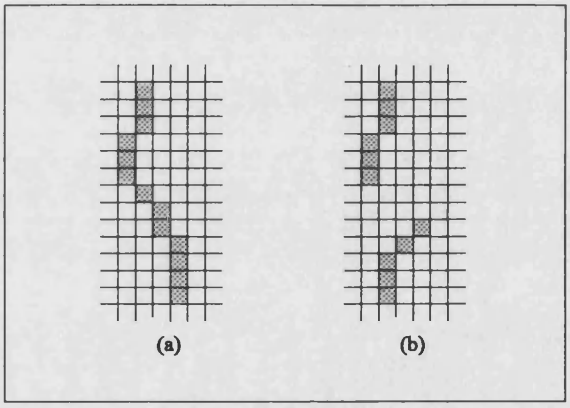


Figure 5.9: (a) A vertical line without discontinuity, each pixel has one neighbour element and (b) a disconnected line.

The responsibility of the filling algorithm is to consider the disconnected area with respect to the disconnected pixel co-ordinates and then place new pixels for sealing the gaps. If a line is broken and needs to be filled between two disconnected pixels, it will be corrected by considering the end pixels based on the *Bresenham* algorithm [96 , 97].

The first point to establish with any line drawing algorithm is which is the independent variable. For example, with a predominantly vertical line, for each 'y' increment there may or may not be a 'x' increment. Also the same effect happens with a line closer to the horizontal. So the independent variable is chosen according to whether the line is closer to the horizontal or vertical.

The *Bresenham* algorithm may be described with the aid of Figure 5.10. Assuming a start from the top, y is incremented by one and x increases by $(y_2-y_1)/(x_2-x_1)$. However the algorithm takes advantage of the incremental nature of pixels and sees this as simply a decision as to whether to keep x the same or increase it.

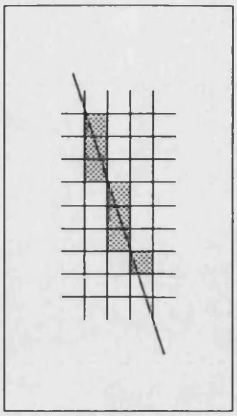


Figure 5.10: Bresenham's line algorithm determines the best fit of pixels

The difference is stored in a variable ε, and $(y_2-y_1)/(x_2-x_1)$ is added for each y increment. If ε exceeds 0.5, then the next row is stepped up to $(x=x+1)$ and 1 is subtracted from ε. *Bresenham* optimised the condition to eliminate the division and use integer math. If the incremental term is multiplied by the y difference (y_2-y_1), and the error is initialised to $-(y_2-y_1)/2$, then only a check for the error exceeding zero is needed. The operation of the filling algorithm is illustrated in an example as shown Figure 5.11.

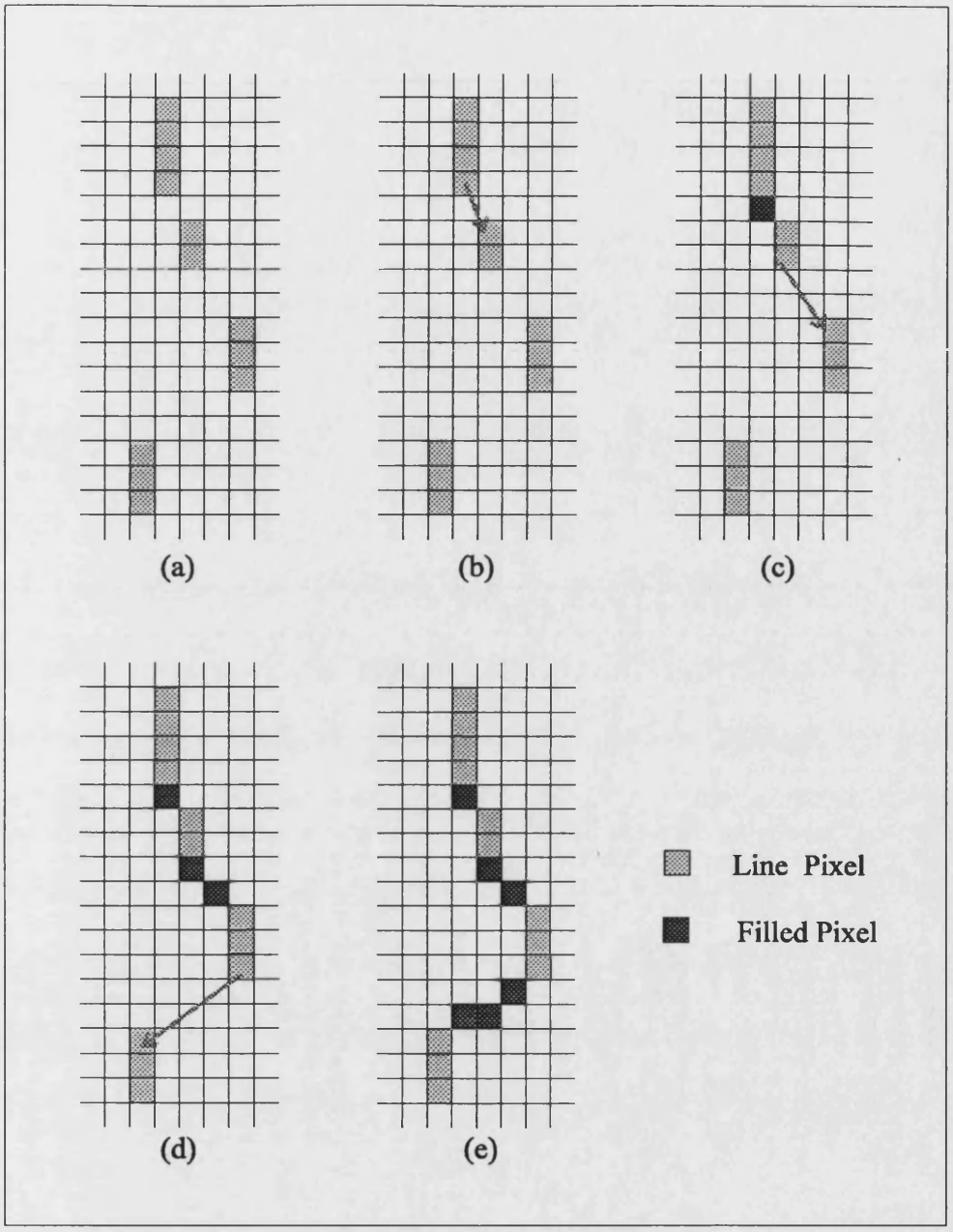


Figure 5.11: Line filling during the forward scanning, (a) a defective line, (b) scanning the line to find the gap and next pixel, (c) filling the gap and continue the scanning, (d) the next gap is revealed, and (e) continue scanning until end of the line.

The tracing and filling algorithms on the reference line will produce unbroken line with two end points indexed as start ($Start_0$) and stop ($Stop_0$) pixels for the reference line. The next stage is applying boundary tracing from the start pixel to find all the border information of the image.

5.2.3 Boundary Tracing

The purpose of the boundary tracing is to locate the extremities of the image surface element. It does so by a common technique of border following [71]. The process starts by extracting a boundary network for the image from the start pixel on the reference line by applying the directional code [98]. The results of the processing are the co-ordinates and directions of the boundary junctions and end points.

The boundary tracing can be started from one of the end points on the reference line, in our application from $Start_0$. The boundary tracing algorithm follows the pixels on the image border from the start pixel as shown in Figure 5.12. The tracing, based on the directions of the neighbourhood pixels, permits the algorithm to follow a continuous boundary. If a gap or discontinuity happens on the boundary, the tracer will go inside the broken area, and scanning will be continued on the boundary pixels until achieving the last pixel on the reference line ($Stop_0$).

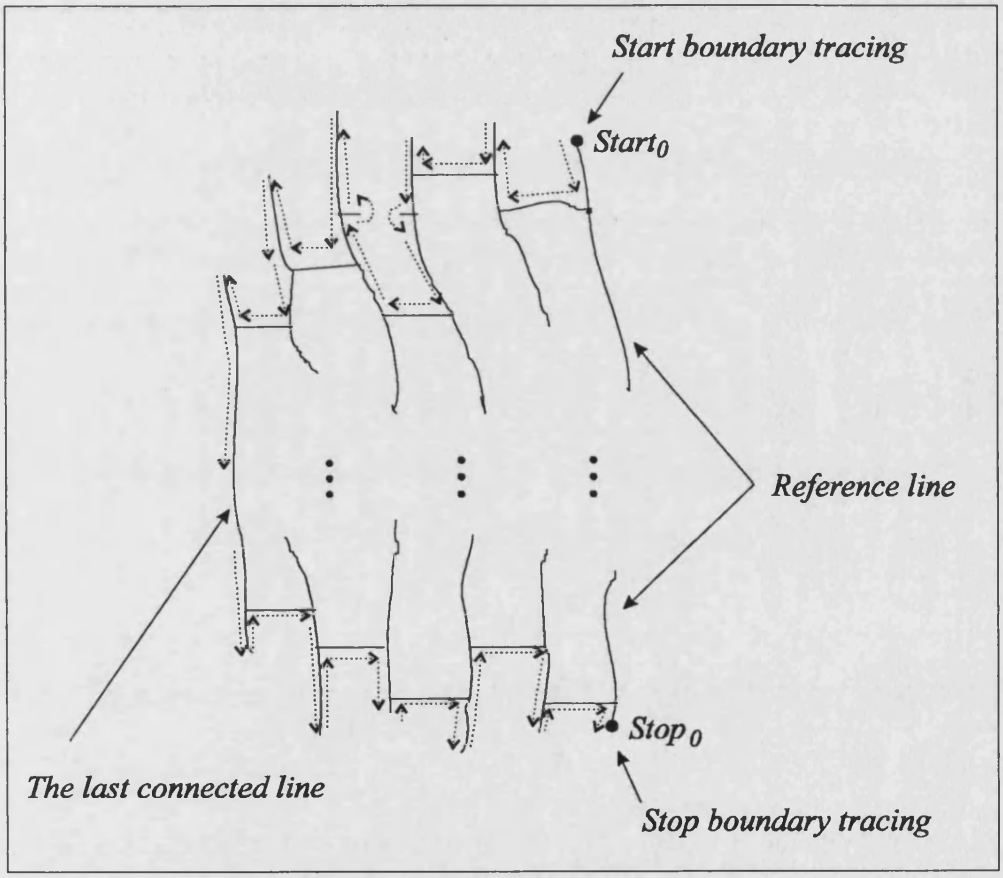


Figure 5.12: Boundary tracing from the start pixel on the reference line

The tracing is implemented by using directional coding such as used in chain code [99]. The basic idea of the chain code is to use only the direction to the next pixel for each of the connected pixels on the boundary [100]. There are eight possible directions for the next pixel neighbours as shown in Figure 5.13, so the direction values can be represented as integers in the range 0-7.

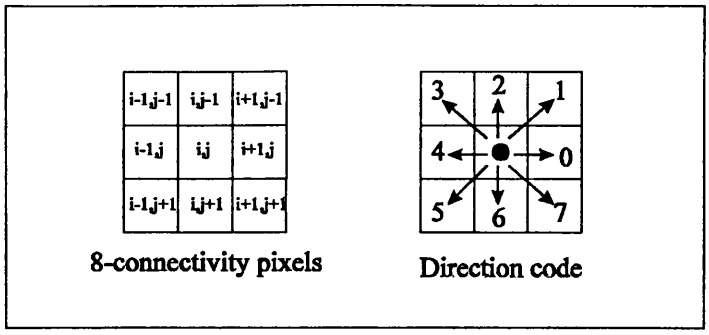


Figure 5.13: 8-connectivity and the directional code (anti-clockwise)

The *kth* contour, S_k, of a binary image can be represented as:

$$\begin{cases} C_k = \{c_0, c_1, \ldots c_i, \ldots c_n\} \\ X_k = \{x_0, x_1, \ldots x_i, \ldots x_n\} \\ Y_k = \{y_0, y_1, \ldots y_i, \ldots y_n\} \end{cases} \qquad \text{Equation 5.1}$$

where C_k is the direction chain code set of contour k, i is the pixel index of the contour, c_o is the starting direction code pointing from the first pixel ($Start_0$) to the second pixel of the contour, c_i is the direction code of pixel i pointing from pixel i to pixel $(i+1)$, c_n is the direction code of pixel n to the last pixel of contour ($Stop_0$). X_k and Y_k are the x and y co-ordinate sets of contour k respectively.

The first step in producing the chain code is to select a starting pixel ($Start_0$). The next step is to locate a neighbour, but just any neighbour is not good enough. It is necessary to follow the boundary always in the same direction. Having chosen to move anticlockwise it is important to select the next pixel with that in mind. The next pixel is found by circling around the current pixel in an anticlockwise direction starting at the previous pixel.

The important note about the directional code is the manner in which the next pixel is located. It is essential to travel from the previous pixel in a consistent rotational direction around the current pixel while searching for the next one. There is a simple way to do this: The last entry in the chain code so far is the direction travelled from the previous pixel to the current. If this direction is called c_i, then the inverse direction (from current pixel to previous) is $(c+4)$. Given this the first place to look for the next pixel will always be in direction $(c+5)$ mod 8 from the current pixel, and proceed in consecutive directions, remembering to wrap around from 7 to 0.

Figure 5.14 shows an example of the boundary and the chain codes generated by the directional coding. The first pixel after start pixel is found by rotating (for our application always anticlockwise) around the start pixel from the direction 0 (or from the neighbour pixel with co-ordinate $(i+1, j)$). The first obtained code is c_1=6. The next step is to locate another neighbour and then repeat it again for all of the boundary pixels. To find the correct direction for starting the rotating for the next code, we need only follow the mentioned instruction. The next pixel is positioned at the direction of c_i+5 (c_1+5=6+5=11 and in mode 8, 11-8=3), therefore the start direction is from 3. By rotating from this direction, the next pixel and next chain code will be found.

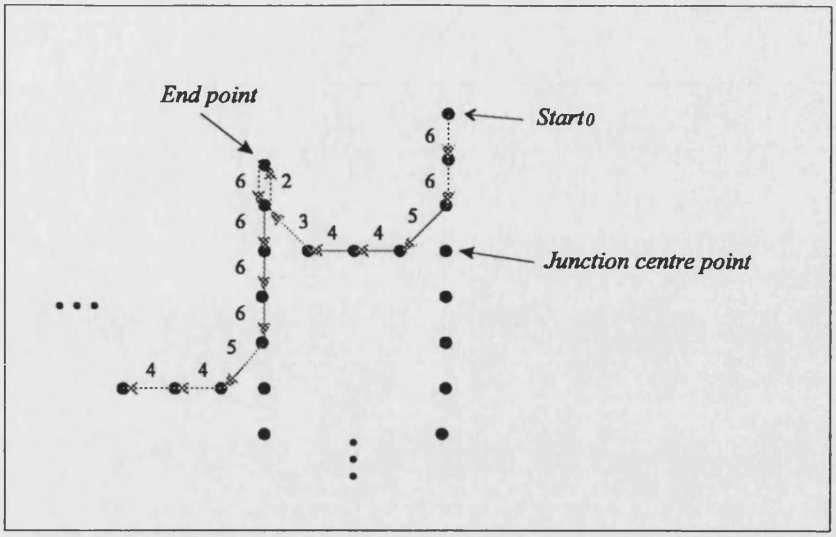


Figure 5.14: Example boundary and its chain code using the directions

The algorithm follows border of the image by applying the directional coding. Co-ordinates of the end points and junctions must be found during the tracing and saved in the boundary network. A junction is known and classified with its centre pixel, the junction point, therefore the co-ordinate of this pixel should be available as well as the end points. The problem with the directional code is that some pixels inside the junctions are ignored during the tracing as shown in Figure 5.14. This problem is compensated by searching inside the 8-connectivity neighbourhood window simultaneously with generating the chain code. When more than two pixels are found in the neighbourhood window, it may be a junction and therefore the algorithm tries to detect it by increasing the window size and testing a series of defined constraints, this is discussed later. Also the direction of end points are important for indexing the vertical lines during the loop tracing stage. All of the above procedures, detection of the junctions (junction centre points (JCP) and direction (a_t)) and end points (type and direction (e_d)), are defined under the title of *classification*.

The classified information obtained during tracing is saved in the boundary network (N_B). The boundary network consists of two arrays for saving the co-ordinates and directions of the end and junction points as defined in equation 5.2. E_B is an array for saving the boundary end points and T_B is another array for saving the boundary junctions.

$$\begin{cases} E_B = \{(x_e, y_e, e_d),\ \text{where } (x_e, y_e) \in \text{Boundary end points and } e_d = 0,\ 1,\ 2 \text{ and } 3\} \\ T_B = \{(x_t, y_t, a_t),\ \text{where } (x_t, y_t) \in \text{Boundary junctions (JCP) and } a_t = 0,\ 1,\ 3 \text{ and } 4\} \\ N_B = \{E_B \text{ and } T_B\} \qquad \text{Boundary network} \end{cases}$$

Equation 5.2

where e_d is the direction of end point and a_t is the junction attribute (discussed later)

The classification of the pixels in the boundary has been implemented by using both the difference code [101] and pixel searching inside the window. The classification of the

image components is done for either boundary tracing or for the next stage, loop tracing.

5.2.3.1 Classification using the difference code

The difference code of a contour is defined as shown in equation 5.3:

$$d_i = c_{i+1} - c_i \qquad \text{Equation 5.3}$$

It can be calculated as,

$$d_i = \begin{cases} c_{i+1} - c_i & \text{if } |c_{i+1} - c_i| < 4 \\ c_{i+1} - c_i - 8 & \text{if } |c_{i+1} - c_i| > 4 \\ 4 & \text{if } |c_{i+1} - c_i| = 4 \end{cases} \qquad \text{Equation 5.4}$$

Therefore, the difference code value is 0, ±1, ±2, ±3 or ±4. If d_i is equal to ±1, ±2 or ±3 then the current direction chain code of pixel i is changed with rotation of ±135, ±90 or ±45 degree. If d_i is equal ±4, the current direction chain code of pixel i is changed with a rotation of 180 degree. In general, most contour pixel direction chain codes are 0 or ±1, otherwise, there should be an end point or a noisy contour pixel. In terms of equation 5.4, all the difference code groups are shown in Figure 5.15.

An end point has only one pixel neighbour so four types of end point may occur in the boundary area (top, bottom, right and left). The difference codes of end points are d_i=±4 as shown in Figure 5.15(f). The direction of each end point is defined as,

(TEP) Top end point (e_d=0): (c_i=2 & c_{i+1}=6), (c_{i-1}=2 & c_i=1 & c_{i+1}=5) or (c_{i-1}=2 & c_i=3 & c_{i+1}=7)
(BEP) Bottom end point: (e_d=1): (c_i=6 & c_{i+1}=2), (c_{i-1}=6 & c_i=5 & c_{i+1}=1) or (c_{i-1}=6 & c_i=7 & c_{i+1}=3)
(REP) Right end point (e_d=2): (c_i=0 & c_{i+1}=4), (c_{i-1}=0 & c_i=1 & c_{i+1}=5) or (c_{i-1}=0 & c_i=7 & c_{i+1}=3)
(LEP) Left end point (e_d=3): (c_i=4 & c_{i+1}=0), (c_{i-1}=4 & c_i=5 & c_{i+1}=1) or (c_{i-1}=4 & c_i=3 & c_{i+1}=7)

Equation 5.5

The direction for each end point (e_d) is a number which is saved in the boundary network. The top end points may be the start pixels for the vertical lines, the bottom end points may be the stop pixels for the vertical lines or they both could be disconnected pixels on the vertical lines. The right and left end points are the disconnected pixels on the horizontal lines. This kind of classification for end points provides necessary information for indexing each vertical line and also permits correction of broken lines on the boundary before starting the loop tracing.

The algorithm can detect a junction by comparing two or more consecutive difference codes. If the values for two consecutive difference codes are d_i & $d_{i+1} \leq -1$, then a junction may be available there. Figure 5.16 shows the directional coding for a junction on the boundary with the consecutive difference code (-1). It is possible that two difference codes with value (-1) are not positioned consecutively in the sequence of the difference codes but they present a junction, e.g. (..., -1, 0, -1,...). Therefore if a difference code with ($d_i \leq -1$) is found, the algorithm will search to find the next negative difference code after one or two difference codes, with values (0), then will assign it as a junction for applying the junction detection algorithm. Detection of a JCP and its attribute will be done by processing the neighbourhood window. The junction attribute is a number for determining the direction of a junction, direct or corresponding (refer to Figure 5.18), which will be used for classifying the junctions. If the junction is a direct junction then the junction attribute will be considered $a_t=1$, and if it is a corresponding junction $a_t=0$.

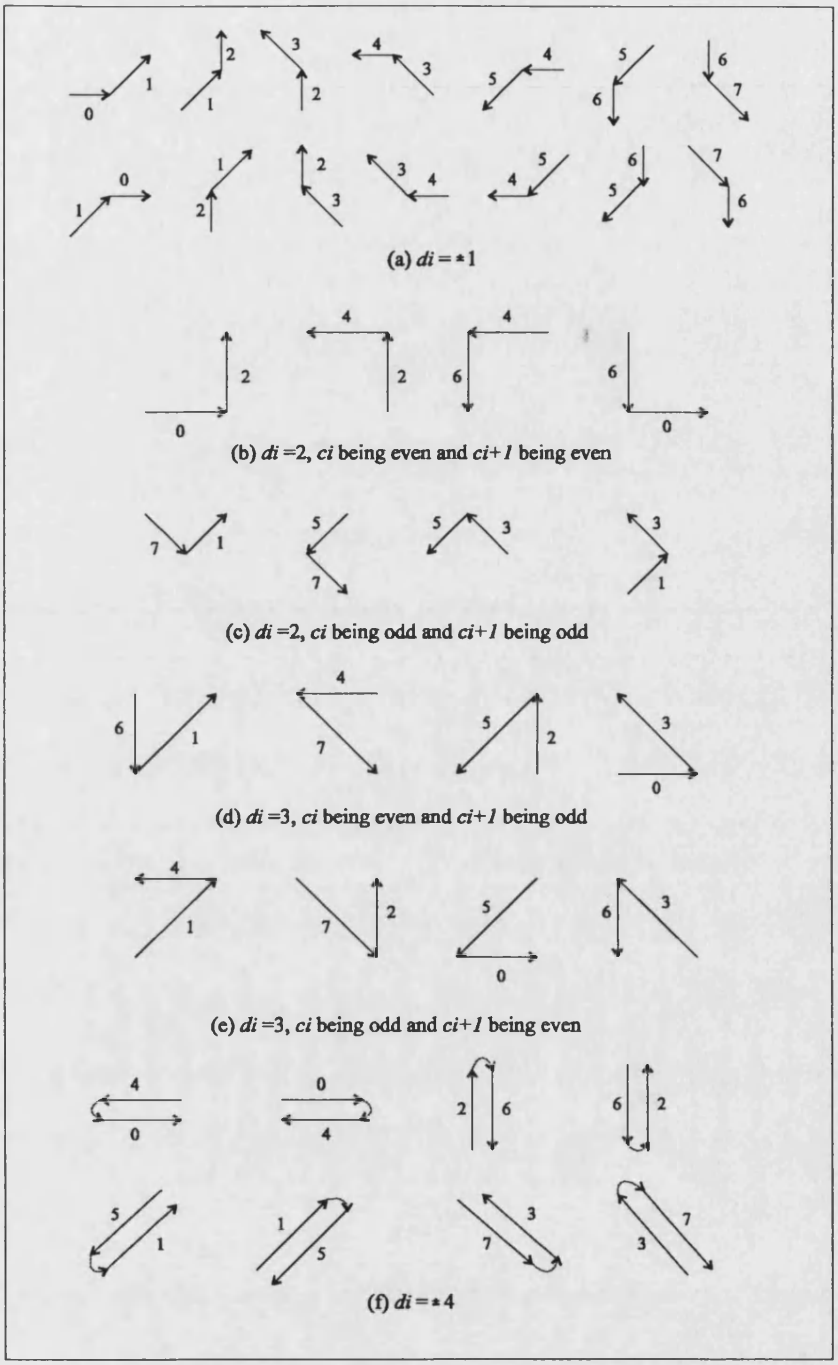


Figure 5.15: Difference code groups

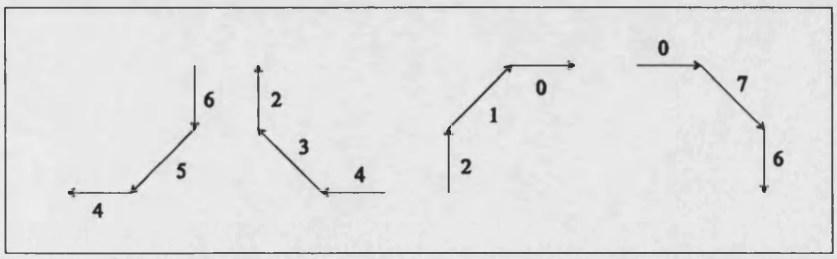


Figure 5.16: Directional coding may happen for a junction when d_i=-1

5.2.3.2 Classification using the neighbourhood windowing

The junction detection algorithm exams the neighbourhood pixels in a defined window to find the condition of the current pixel. During tracing, the boundary pixels are classified in different groups (an end point, a connected pixel of a line or a junction) and a degree will be assigned for each group. For each pixel on the border its degree (of value 1, 2 and 3) is computed by counting the number of white pixels in 3×3 neighbourhood. Pixels of degree-1 are end points. Pixels of degree-2 are connecting points on lines and degree 3 points may be a junction as shown in Figure 5.17.

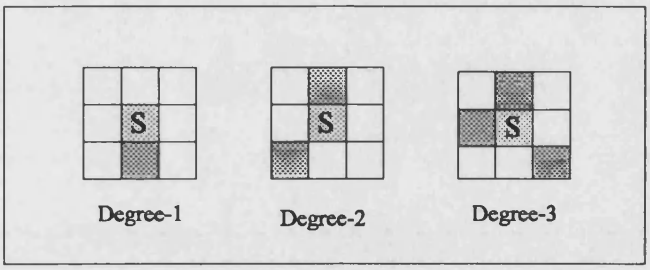


Figure 5.17: Degree identification in 8-connectivity

Detecting the degree-1 points, defined before as end points, have been done using the difference code. The purpose of this section is to find and classify the junction centre points (probably degree-3). Since the feature extraction stage sometimes generates degree-4 or 5 for a junction, an algorithm is applied on the candidate pixel (when its degree is ≥ 3) by increasing the windowing radius from 1 to 3 pixels. If all the defined conditions are satisfied by the candidate pixel, the window centre will be taken as the T-junction centre point (JCP) and then will be saved in N_B.

Algorithm (5.2.3.1) to detect a T-junction and its centre (JCP):

T-junctions are important features for our image analysis. The proposed method is an extended technique for the gray level image junction detection methods that were introduced by *Noble* [102] and *Parida et al.* [103]. Detection is based on a set of conditions on the set of nonzero pixels lying in a square neighbourhood centred about each candidate pixel. The candidate pixel for applying the algorithm is the degree-3 pixel (or greater) with co-ordinates (x, y) which is found during the tracing. Figure 5.18 shows a 7 by 7 square neighbourhood centred about the candidate (i.e., centre) pixel. The nonzero pixels are shaded in gray and the border pixels are marked by an 'X'.

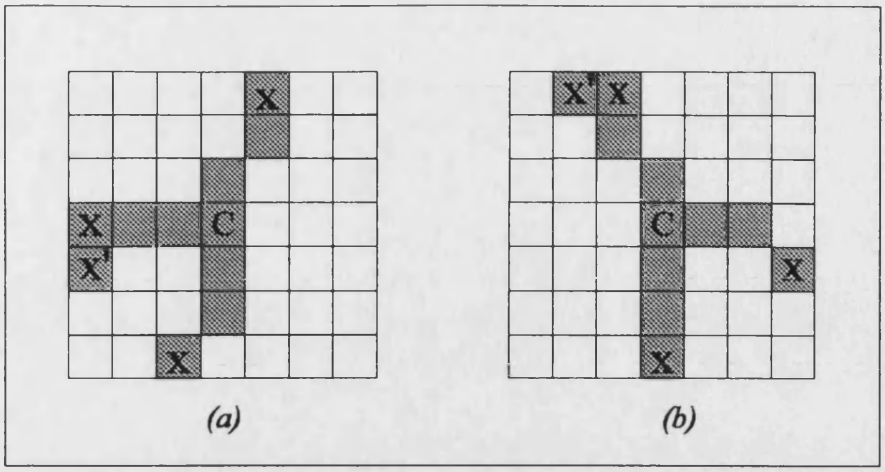


Figure 5.18: T-junction detection by a 7 by 7 window, (a) direct and (b) corresponding junctions. The window centre is positioned on the candidate pixel (C) and the border pixels are shown with (X).

The detection of a junction and its centre is done by applying the following algorithm along the image during the tracing.

1. For each pixel on the boundary during directional tracing determine its degree by counting its neighbouring pixels in an eight connectivity area. If a degree-3 pixel is found, index it as a candidate pixel and follow the algorithm.

   Figure 5.19 shows that the degree of each connected pixel to the junction is 3 or greater than 3. Therefore if a pixel with a degree ≥ 3 occurs during the directional tracing, the algorithm starts detecting the junction and its centre pixel. For a direct or corresponding junction when the JCP is positioned exterior to the contour (a, b, c and d), the candidate pixel for JCP can be found from the common pixels between two windows on the consecutive pixels by considering the equation 5.6. But if the JCP is positioned on the traced contour (e and f), it can be detected easily. Also some noisy pixels on the line generates degree-3 points (g) and will be pruned during the process (stage 3) .

   For four consecutive pixels during the boundary tracing, p_i, p_{i+1}, p_{i+2} and p_{i+3},

   If $[(d_i \,\&\, d_{i+1}=-1)$ or (degree $(p_{i+1} \,\&\, p_{i+2}) \geq 3)]$ Equation 5.6

   Then consider the exterior common pixel of two windows around the p_{i+1} and p_{i+2}, or the co-ordinates of the cross point of the directions c_i and c_{i+2}, as the candidate pixel.

   If $[(d_i \,\&\, d_{i+1}\neq-1) \,\&$ (degree $(p_i \,\&\, p_{i+1}) \geq 3)]$ Equation 5.7

   Then consider pixel p_{i+1} as the candidate pixel.

   Besides of the above conditions, a start candidate pixel may be selected from a degree-3 point or a pixel with difference code (-1). If the selected pixel is not an

appropriate point for the junction centre, it will be replaced with a suitable one during the junction detection (stage 7).

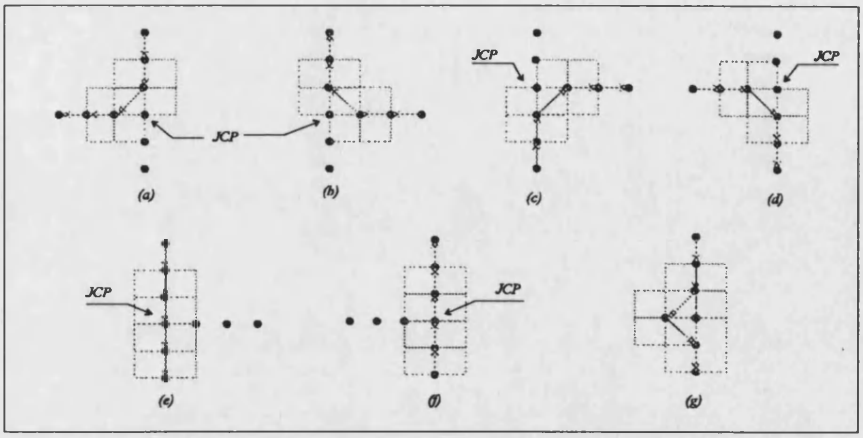


Figure 5.19: Detection the candidate pixel for applying the junction algorithm, (a, b, c and d) JCP is exterior to the contour, (e and f) JCP on the traced contour and (g) a noisy pixel on the line.

2. Increase the window size from (3×3) to (7×7) around the candidate pixel.

3. Check the number of the nonzero border pixels in the square neighbourhood centred on the candidate pixel (X). Minimum of three connected subsets are necessary for each T-junction. If the number of border pixels is less than 3, decrease the window radius to 2 (5×5) and check the number again. If the number of border pixels is still less than three, mark the candidate pixel as a degree-2 and apply pruning then continue tracing, if not continue the process with the new window (5×5).

4. Find the correct border pixels by calculating the length of the shortest paths connecting the centre candidate pixel with each of the border pixels (X_{top}, X_{bottom}, X_{right} or X_{left}). Each thinned stripe may touch the side of the square at more than one point as shown in Figure 5.18, the border pixels with the shortest paths are shown with (X). Two paths from the border pixels must be available, one from X_{top} to X_{bottom} and another from X_{right} (or X_{left}) to the candidate pixel, and the candidate pixel required to lie on each of these paths.

5. Check the position of the border pixels and the angles of the two paths. X_{top} should be on the top row border, X_{bottom} on the bottom row border and X_{right} on the right column border (X_{left} on the left column border). The angles of each path are calculated from the horizontal axis with centre of the candidate pixel. The calculated angle is compared with 0 and ±45 degree for each path to find a rotated or distorted junction.

6. When all of the conditions are met, assign the candidate pixel as the junction centre point (JCP). Detect the junction attribute (direct or corresponding junction) by considering the X_{left} or X_{right}, and save the co-ordinates and attribute of JCP in N_B.

7. When the conditions are not met for the candidate pixel, assign a new candidate pixel by finding the cross point of two paths from the border pixels, as explained in stage 4. If there is not a pixel at this co-ordinate, insert a pixel as a candidate pixel and continue the process.

8. Apply the junction pruning and determine the line crossing condition. A practical note about the junctions are some noisy pixels connected to them after the feature extraction stage. This problem produces unpleasant and unwanted pixels around the candidate pixel as shown in Figure 5.20 (a). The junctions have been evaluated by windowing and the stage 4, and then unnecessary pixels are removed from the main structure.

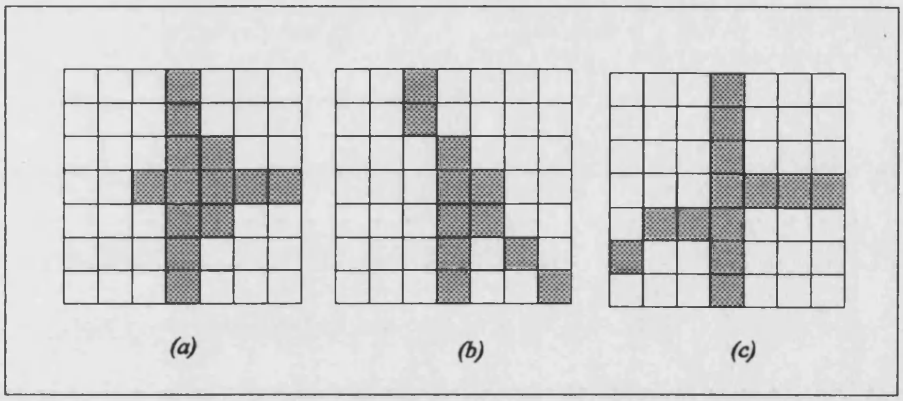


Figure 5.20: Different conditions may occur during the junction detection algorithm, (a) noisy pixels around a corresponding junction which will be pruned, (b) three connected paths to the candidate pixel but the conditions of a junction are not satisfied and (c) cross point of four paths to the candidate pixel.

Pruning a junction must be applied carefully because some pixels around the candidate pixel provide the information of the overlapped or connected lines which will be used later, the loop tracing stage, for classifying the image components. Figure 5.20 (b) shows that three paths are available for indexing a junction but they can not satisfy the conditions of a junction. Also Figure 5.20 (c) shows that four paths from the border pixels are available. The both conditions are important and therefore the pixels of each path are remained and only the noisy pixels around the lines in the window will be removed. The candidate pixel is then indexed by a number (a_t=3 or 4), which defines the number of cross lines from the candidate pixel to the borders, and the co-ordinate of the candidate pixel and its attribute will be saved in the junction array.

5.2.3.3 Ambiguity of a junction or line

In a stripe network, points of degree 1 and degree 3 are easy to handle, because degree 1 points are always on network borders, and the degree 3 points are always interior to the network and will be processed by the junction detection algorithm. So we only need to worry about points of degree 2, which may be a distorted junction and end points and therefore it is difficult to get their order correct on the boundary as illustrated in Figure 5.21.

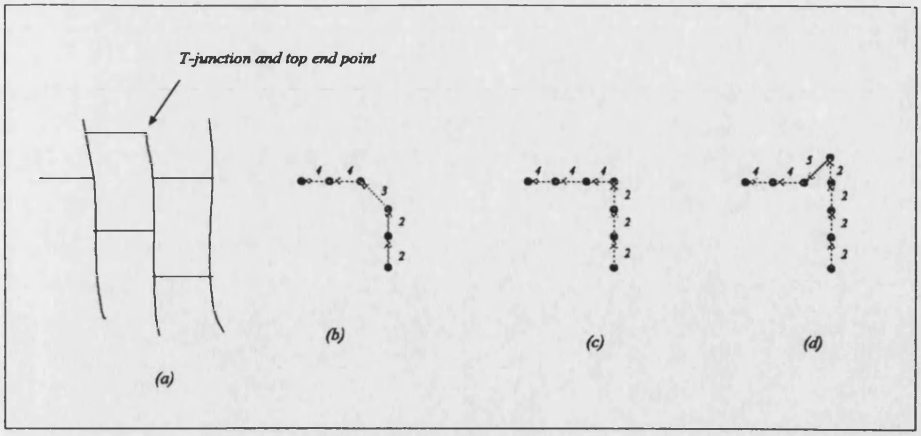


Figure 5.21: The order of points on boundary may not be correct, a top end point and a direct junction overlapped. (a) A junction and end point are at the same place, (b) problem from a degree-2 pixels with d=1, (c) problem on degree-3 pixels with d=2 and (d) problem on degree-3 and d=3.

During the tracing when two consecutive points with degree-2 and d_i=d_{i+1}=1 is recognised, the algorithm extends its degree to a degree-3 as follow. For each image point of degree-2 (where a horizontal stripe and a vertical stripe meet but do not make a T-junction, i.e. a corner), the algorithm extends the degree to 3 by adding a pixel as a junction point (p) and a degree-1 point to the y-stripe. This means that we add a point p_1 to p such that p_1 is only one pixel off p and in the direction extended by the y-stripe edge of p as shown in Figure 5.22.

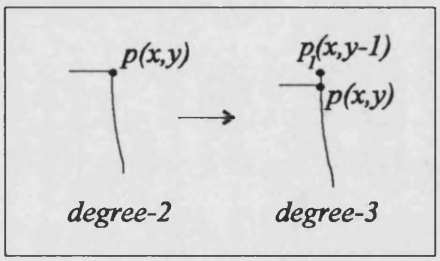


Figure 5.22: Extending the degree 2 point to the degree 3, by adding a point (p) and end point (p_1) to the y-stripe.

For a degree-3 point with d_i=2 (Figure 5.22(b)), the junction point is available and the algorithm needs only to add a p_1 off the p the same procedure as above. For a degree-3 with d=3, the algorithm saves the junction point and the top border pixel of the junction as the end point.

For other ambiguities of the junctions and lines on the boundary as shown in Figure 5.23, a top end point and corresponding junction and a bottom end point with both direct and corresponding junctions, the explained procedure will be repeated by considering the direction of y-stripe for p_1.

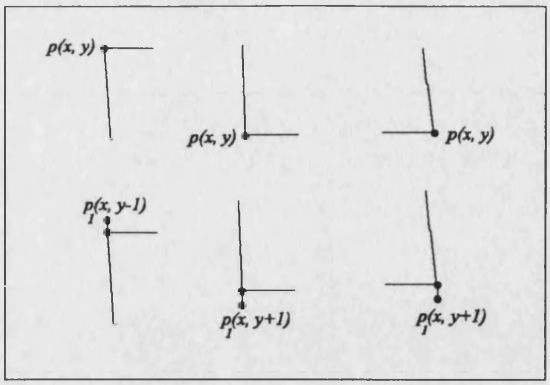


Figure 5.23: Other ambiguities for a junction and a line and adding an end point p_1 to the y-stripe.

5.2.3.4 Boundary correction

The last stage is to complete any small discontinuities on the boundary after finishing the tracing. The tracing algorithm scanned the boundary from the start pixel ($Start_0$) and saved the boundary information in the boundary network. If an unpredictable end point is seen in the boundary network, the algorithm will try to find and correct the problem. This can be done by searching the end points and their direction in the boundary network. Two examples of unwanted end points in the boundary network are shown in Figure 5.24. TEP is a top end point, BEP a bottom end point, REP a right end point and LEP a left end point.

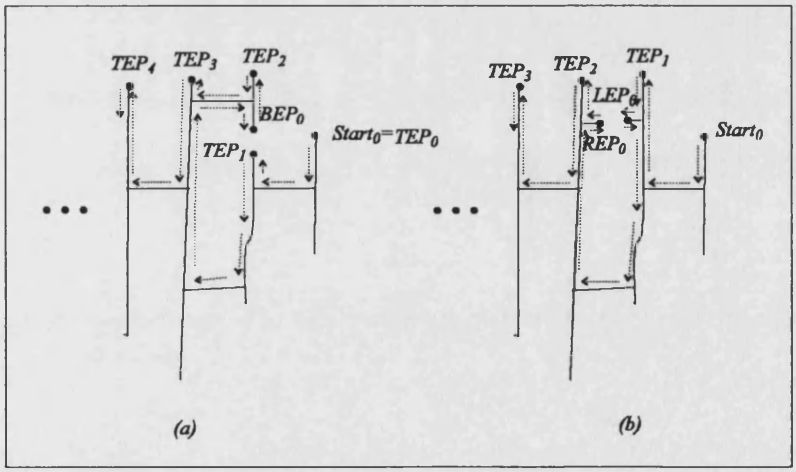


Figure 5.24: Discontinuity on a (a) vertical or (b) horizontal line on the image border which may cause unwanted end points on the boundary.

In condition (a), a bottom end point (BEP_0) is available in a series of top end points so that it must be a disconnected vertical line. The algorithm connects BEP_0 to the previous end point (TEP_1) using the filling algorithm. In condition (b), two unwanted end points (LEP_0 and REP_0) occur in a series of top end points which are belong to a horizontal line and therefore the algorithm can connect them easily. Also the same conditions can be explained among a series of BEPs.

The process of correcting the boundary will be continued until an equal number of top and bottom end points is achieved. This means that for each top end point we have a corresponding bottom end point. If (TEP_0, TEP_1, ..., TEP_N) are the series of top end points and (BEP_0, BEP_1, ..., BEP_N) the series of bottom end points, then each pair of end points will be defined for a vertical line as follow,

$$\{(TEP_0, BEP_N), (TEP_1, BEP_{N-1}), ..., (TEP_{N-1}, BEP_1), (TEP_N, BEP_0)\} \quad \text{Equation 5.8}$$

where N is the total number of vertical lines (TEP_0=$Start_0$ and BEP_N=$Stop_0$)

It is clear that the algorithm can not correct all of the right or left end points which may occur in the sequence because a pair point is not necessarily available for all of them in the boundary, refer to Figure 5.3.

After classifying all the boundary components and saving their co-ordinates in the boundary network, the co-ordinates of the top end points and junction points are used during the loop tracing stage. A complete reference line and boundary network provides all the necessary information about the image border and now the loop tracing algorithm can scan the image.

5.1.4 Loop Tracing

The goal of the loop tracing is to scan, correct and label all of the loops inside the extracted image by considering the boundary information from the previous stages and finally prepare the image for the reconstruction stage. A loop here is defined as a closed contour consisting of two approximately vertical and two approximately horizontal lines containing four T-junctions in the corners and two T-junctions between them as shown in Figure 5.25. The tracing algorithm scans inside the loops to find all the above junctions. If any problem occurs inside the loops, the algorithm will find and compensate it by extending the scanning area and classifying all the information from the loops.

As the loop tracing can be defined as a contour tracing inside the loop, the problem should be overcome by finding an appropriate method for tracing. Different approaches have been introduced for contour tracing such as hierarchical [104], piecewise linear approximation [105], forward and backward [106] and dynamic programming [107]. Most of the proposed techniques for contour tracing make use of the directional code or region adjacency graph for achieving the contour information [108 , 109]. As the directional coding algorithm is implemented for the boundary tracing, it will be convenient if the loop tracing algorithm can also employ this technique.

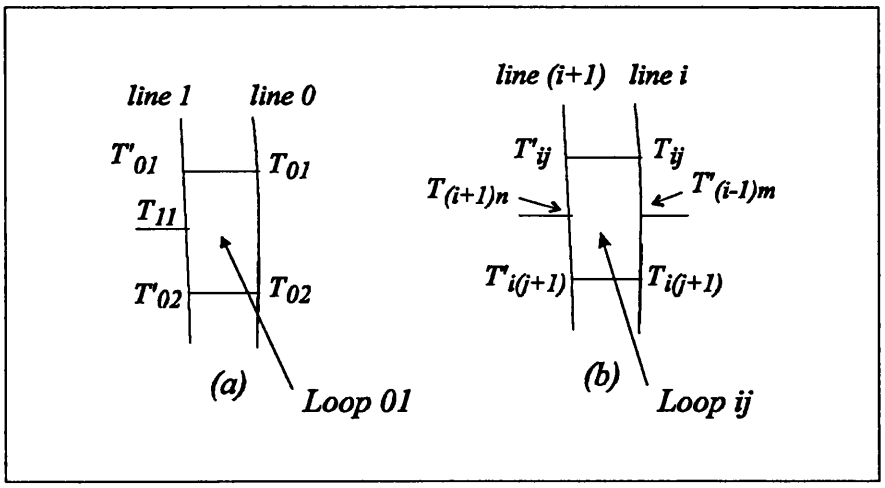


Figure 5.25: (a) The first loop on the reference line, (b) a loop on the line *i*

The components of a loop are;

- Direct junctions on the line i, (T_{ij}, $T_{i(j+1)}$).
- Corresponding junctions on the next line i+1, (T'_{ij}, $T'_{i(j+1)}$).
- Median junctions on line i ($T'_{(i-1)m}$) and the next line i+1 ($T_{(i+1)n}$). Each loop on the reference line has only one median junction on the next line.
- The median junction on line i is a corresponding junction for line i-1 and the median junction on line i+1 is a direct junction for that line.

One of the special characteristics of the implemented pattern in comparison with a simple grid pattern is the availability of two extra median junctions between the direct

and corresponding junctions that provide more information for classifying the junctions. Each loop is defined with the first T-junction and surrounded by six neighbour loops, except the reference line and boundary loops, as shown in Figure 5.26.

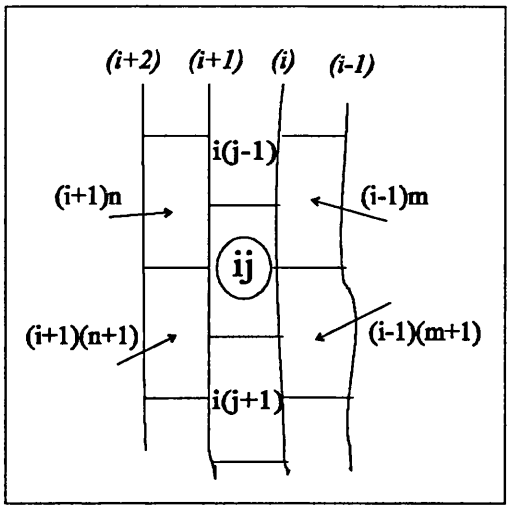


Figure 5.26: Loop *ij* and its neighbours

5.1.4.1 Loop tracing by inverse directional coding

The loop tracing is done by using a directional neighbourhood searching method by considering the junctions and discontinuities inside the loop. The tracing method inside the loop is the same as that used for the boundary tracing but instead of the directional code, an 'inverse directional code' (IDC) has been proposed. The structure of the inverse directional coding is shown in Figure 5.27(a). If c_i is the direction code of pixel i to pixel $i+1$, c'_i is defined the inverse code of pixel i to pixel $i+1$ in the anti-clockwise direction where,

$$c'_i = 4 + c_i \qquad \text{Equation 5.9}$$

and also inverse difference code is,

$$d'_i = c'_{i+1} - c'_i \qquad \text{Equation 5.10}$$

From the equation 5.9, all of the conditions for the inverse difference code are the same as explained before for the difference code (Figure 5.15) only the direction of the arrows must be inverted.

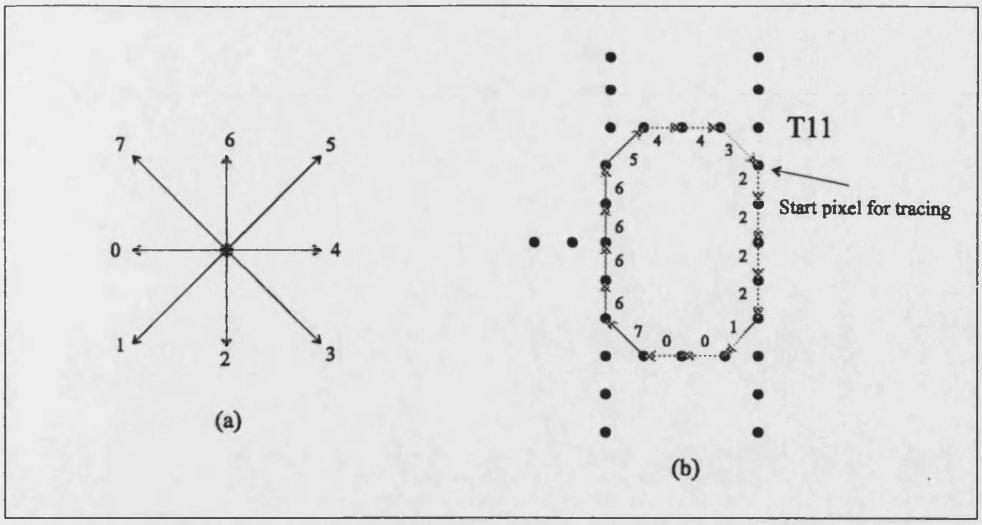


Figure 5.27: (a) The inverse directional code (IDC), and (b) how IDC can trace the loop.

Figure 5.27(b) shows the loop tracing using the inverse directional code which started from the first direct junction on the line. The start point for tracing may be selected from one of two junction pixels inside the loop ($X`_{left}$ or $X`_{bottom}$ from Figure 5.28). But to ensure that the tracing is carried out inside the loop, the algorithm selects the bottom border pixel of the junction ($X`_{bottom}$) by applying a 3 by 3 window around the JCP.

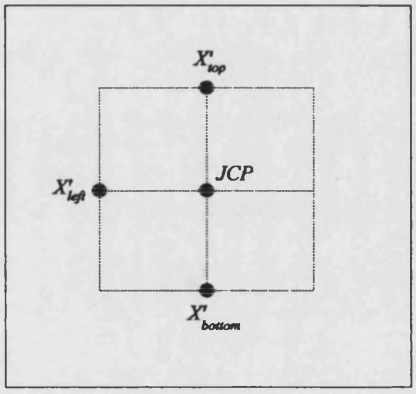


Figure 5.28: Loop tracing starts from the $X`_{bottom}$ pixel of a direct junction.

The loop tracing algorithm starts from the first junction on the reference line, closest to the start pixel, and continues on the loops that lie on the reference line. The loop tracing ends on the last junction, the nearest to the stop pixel, on the reference line when all the loops on the line have been scanned and corrected. The process will be continued for the loops on the second line and then line by line to the last line on the face. During tracing junctions are identified by applying the windowing of neighbourhood, algorithm 5.2.3.1, on each tracing pixel. For the junctions inside the loop, the consecutive inverse difference codes are (-1) as shown in Figure 5.29. At this condition, the JCP is positioned exterior of the tracing contour, but the JCP for the median junctions usually is on the tracing contour.

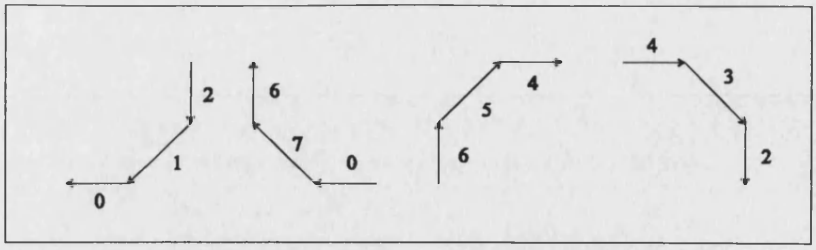


Figure 5.29: Inverse directional coding for the junctions inside the loop

When a loop is broken and therefore more loops are involved, the tracing algorithm scans inside of the broken loops until a close contour achieved. At this condition, a number of disconnected points (end points) will be produced because of the disconnected lines and these are identified using the inverse difference code. The inverse difference codes of end points are $d'_i=\pm 4$ as shown in Figure 5.30. The direction of each end point is defined the same condition as the end points on the boundary,

(TEP) Top end point ($e_d=0$): ($c`_i=6$ & $c`_{i+1}=2$), ($c`_{i-1}=6$ & $c`_i=5$ & $c`_{i+1}=1$) or ($c`_{i-1}=6$ & $c`_i=7$ & $c`_{i+1}=3$)
(BEP) Bottom end point ($e_d=1$): ($c`_i=2$ & $c`_{i+1}=6$), ($c`_{i-1}=2$ & $c`_i=1$ & $c`_{i+1}=5$) or ($c`_{i-1}=2$ & $c`_i=3$ & $c`_{i+1}=7$)
(REP) Right end point ($e_d=2$): ($c`_i=4$ & $c`_{i+1}=0$), ($c`_{i-1}=4$ & $c`_i=5$ & $c`_{i+1}=1$) or ($c`_{i-1}=4$ & $c`_i=3$ & $c`_{i+1}=7$)
(LEP) Left end point ($e_d=3$): ($c`_i=0$ & $c`_{i+1}=4$), ($c`_{i-1}=0$ & $c`_i=1$ & $c`_{i+1}=5$) or ($c`_{i-1}=0$ & $c`_i=7$ & $c`_{i+1}=3$)

Equation 5.11

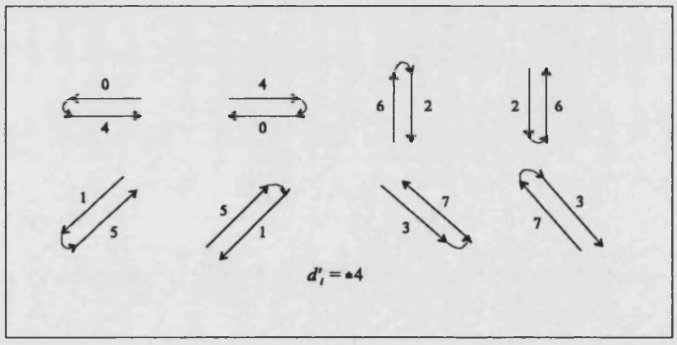


Figure 5.30: Inverse directional codes for the end points

The directional code is very useful in finding and classifying the junctions and end points inside the broken loops but can not detect the median junctions because they are positioned outside of the tracing area. Therefore, simultaneously with difference code extraction, a windowing procedure, such as explained for the boundary tracing, has been used to identify the T-junctions and their sequences.

5.1.4.2 Data structures for loop tracing

All the information obtained during tracing is saved in a series of arrays for processing on the loops as,

(I). Permanent arrays: *Junction Labelled Arrays* (JLA) are the arrays for saving all the labelled junctions (JCP for the direct and its corresponding junction) on the image after processing their loops. For example when the tracing algorithm scans the loop (*ij*) from the junction T_{ij}, the co-ordinate and line number of this junction will be saved in the *Direct Junction Labelled Array* (DJLA) and the co-ordinate and line number of its corresponding junction ($T^{`}_{ij}$) will be saved in the *Corresponding Junction Labelled Array* (CJLA). The arrays are defined in equations 5.12 and 5.13.

$$\mathrm{DJLA} = \left\{ \begin{array}{l} (x_D, y_D, e_D), \text{ where } (x_D, y_D) \text{ is coordinate of the start junction for each} \\ \text{loop (JCP of the direct junction) and } (e_D) \text{ is the vertical line number} \end{array} \right\}$$

Equation 5.12

$$\mathrm{CJLA} = \left\{ \begin{array}{l} (x_C, y_C, e_C), \text{ where } (x_C, y_C) \text{ is coordinate of the corresponding junction} \\ \text{for each start junction and } (e_C) \text{ is the vertical line number } (e_C = e_D + 1) \end{array} \right\}$$

Equation 5.13

(II).Temporary arrays:

(II.1) *Loop network array* (N_L): N_L includes the *loop end points* (E_L) and *loop junctions* (T_L) arrays.

$$\left\{ \begin{array}{l} \mathrm{E_L} = \{(x_e, y_e, e_d), \text{ where } (x_e, y_e) \in \text{loop end points and } e_d \text{ is end point direction}\} \\ \mathrm{T_L} = \{(x_t, y_t, a_t), \text{ where } (x_t, y_t) \in \text{ loop junctions (JCP) and } a_t \text{ is junction attribute}\} \\ \mathrm{N_L} = \{\mathrm{E_L} \text{and } \mathrm{T_L}\} \qquad\qquad \text{loop network} \end{array} \right.$$

Equation 5.14

where e_d (0, 1, 2 and 3) is the direction of an end point (TEP, BEP, REP and LEP) and a_t (0, 1, 3 and 4) is the attribute of a corresponding junction, direct junction, a cross point with three lines and four lines. The total number of end points and junctions inside the tracing loop are defined as n_e and n_t. For a complete loop, we have n_e=0 and n_t=6.

(II.2) Difference code sequence array (DCS): This is an array of the inverse directional coding ($d^{`}_i$) which will be used for saving the all possible difference codes inside the loops and from there identifying the end points and their directions.

DCS={*The difference code sequences from the start point for the loops*} Equation 5.15

The DCS describes the different situations that may be occur during the loop tracing. For example from Figure 5.27 we can write the difference code sequence as,

DCS={0-0-0-(-1)-(-1)-0-(-1)-(-1)-0-0-0-(-1)-(-1)-0-(-1)-(-1)}

The underlined numbers with value (0) show a vertical or horizontal line and the underlined numbers (-1) are the junctions inside the loop. The difference code for any disconnected point inside the loop will be (±4). We know that different conditions are available on the lines because of turning condition or noisy pixels as shown in Figure 5.31, but it is not possible for two difference codes (greater than 1) and the same polarity to occur consecutively for a line. Therefore besides of the end points, the internal junctions of the loops can be detected by DCS.

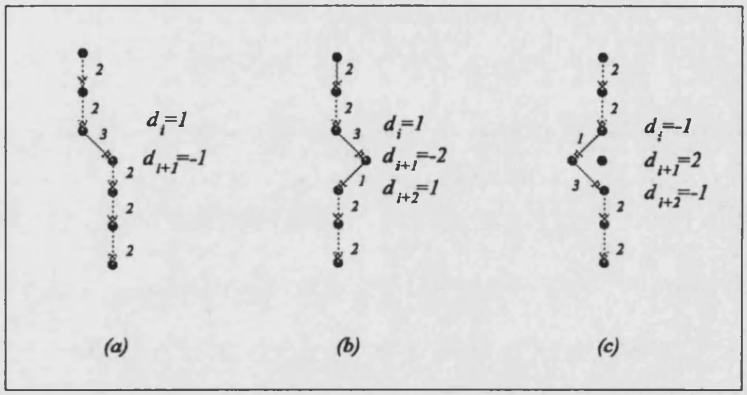


Figure 5.31: Possibilities of the different difference codes on a line. (a) Turning condition for the line, (b) changing the position of a pixel on the line, and (c) a noisy pixel on the line.

From the loop network arrays, another important sequence can be defined which will be used for processing the loops and their deficiencies as,

JS={*Junction sequences of the loops from the start junction*} Equation 5.16

The JS (Junction Sequence) gives the sequence of direct, median and corresponding junctions inside the loops detected during tracing by the junction detection algorithm (5.1.3.1). For a complete loop as shown in Figure 5.25(b), the sequence of junctions can be written as,

$\{T_{ij}\text{-}T\hat{}_{(i-1)m}\text{-}T_{i(j+1)}\text{-}T\hat{}_{i(j+1)}\text{-}T_{(i+1)n}\text{-}T\hat{}_{ij}\text{-}\ T_{ij}\}$

The co-ordinates of the junctions are saved in T_L therefore JS can be written in a simple form,

$JS=\{T_1\text{-}T\hat{}_2\text{-}T_3\text{-}T\hat{}_4\text{-}T_5\text{-}T\hat{}_6\text{-}T_1\}$ Equation 5.17

where T is a direct junction and $T\hat{}$ is a corresponding junction and each subscript number shows the junction sequence. A complete loop has six junctions (the number of junctions in JS) and all the sequences are correct ($T\text{-}T\hat{}\text{-}T\text{-}T\hat{}$-...).

The defined arrays are temporary arrays only for processing inside the traced loops and will be deleted after finishing the process.

5.1.4.3 Loop Tracing Algorithm

The overall structure of the loop tracing algorithm is defined here;

1. From the boundary network find the first direct junction (T_{01}) on the reference line from the start pixel ($Start_0$).
2. Save the start junction (JCP of T_{01}) and line number in the direct junction labelled array (DJLA).
3. Apply the 8-connectivity window around the JCP to find the start pixel for tracing ($\hat{X}_{bottom}$).
4. Scan inside the first loop from the first junction (T_{01}) by considering the inverse directional coding (IDC).
5. Detect the junctions (algorithm 5.1.3.1) and end points inside the tracing area.
6. Save the loop information in the loop network (N_L), and the difference code sequence (DCS).
7. Search for the correct sequence of the junctions inside the above arrays, (1) start junction (T_{01}), (2) next direct junction of the line (T_{02}), (3) corresponding junction ($\hat{T}_{02}$), (4) median junction (T_{11}), (5) corresponding junction ($\hat{T}_{01}$) and (6) start junction again (T_{01}). (This is a sequence for the loops on the reference line)
8. If the sequence is correct, this means that the loop is complete (problems within the loop are discussed later), save the corresponding junction of the start junction (*JCP* of $\hat{T}_{01}$) in the corresponding junction labelled array (CJLA) and then delete the temporary arrays (N_L and DCS).
9. Go to the next junction on the reference line (T_{02}), the first direct junction on the line after the start junction, and continue tracing for the next loop (stages 2-8)
10. Continue tracing (stages 2-9) until reaching the last junction and end pixel (BEP) for the processed line ($Stop_0$) on the boundary network (N_B).
11. Go to the second line by finding the next start pixel in the boundary network and repeat the stages (2-10) from the first junction on this line.
12. Continue the above stages (2-11) for all the lines and their loops until achieving the last complete loop on the last line.

The above procedure is an overview of the loop tracing algorithm. But a real face makes different problems on the loops by breaking or merging of the projected lines. Consider that the loop *ij* and its neighbours are broken. The loop tracing starts from T_{ij} and scans the pixels inside the closed contour (the broken loops) by IDC. The loop tracing stops on achieving T_{ij} again. After saving the information of the traced contour in the defined arrays (N_L, DCS), the problem is classified and a general solution will be

applied for recovering the area. The structure of the proposed algorithm can be improved by considering the following stages,

a) Save the current junction (T_{ij}) as a pointer and then follow the broken path inside the other broken neighbour loops.

b) Save all the information from the junctions and end points inside the broken area in the loop network (N_L).

c) Generate the difference code and junction sequences along with stage (b).

d) Classify the problem area by considering the relative information in the sequences, the number of junctions and end points.

e) Determine the missing components inside the problem area such as lines or junctions based on the classification for a small area of discontinuity, discussed later.

f) Correct the disconnected loops for a large area of discontinuity only for the first traced line and based on the classification of the end points.

g) Determine the missing junctions, direct and corresponding, on the filled line using the relative junctions in the junction sequence.

h) Apply the filling algorithm for the stages e, f and g if necessary.

i) Delete the temporary arrays and go back to the pointer junction (T_{ij}) and continue the loop tracing on the line.

The loop tracing algorithm is summarised in Figure 5.32. The first junction for each line is determined by reference to the boundary network, where the line end points and start junction has been saved.

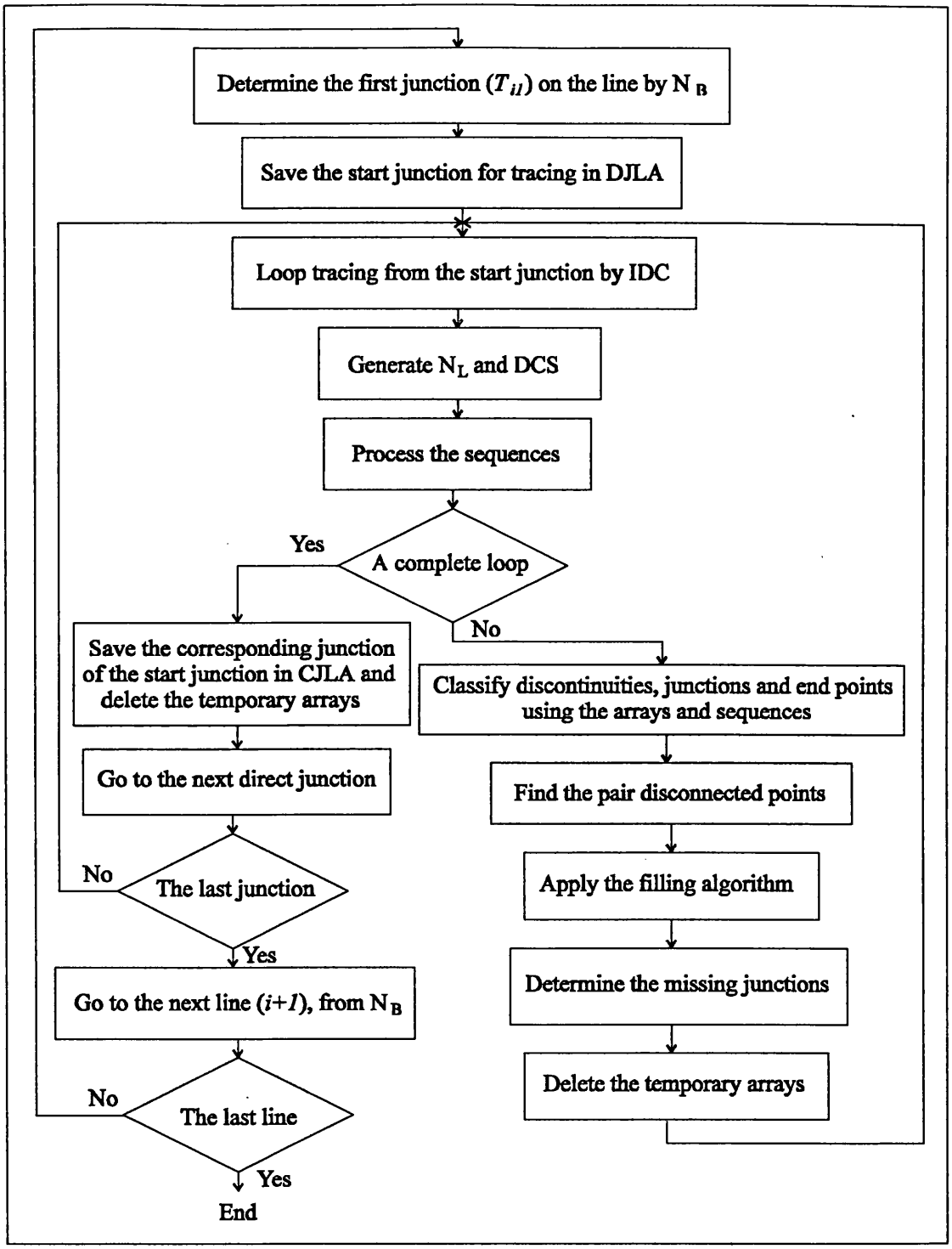


Figure 5.32: Loop tracing stages

The above stages are the main structure for processing the lines and their loops on the image. The next stage is to classify the problems on the scene and then to see how the saved sequences can solve these problems. Discontinuity on a line or a junction are the simple form of the problems in the loops, but missing and overlapping of two or more junctions and lines frequently occur in the facial image. These problems are considered next with a general method for recognising the deficiencies and how to fix them.

5.1.4.4 Discontinuities on the lines

a) One discontinuity on a line

The simplest discontinuity may occur on the vertical or horizontal lines when all the junctions are available in the loops as shown in Figure 5.33. For these type of discontinuities only two loops are involved. The difference code and junction sequences are as follows,

(a)

DCS={...-(4)(E_1)-...-(-4)(E_2)-...} Equation 5.18

JS={T_1-$T`_2$-$\underline{T_3}$-$\underline{T_4}$-$T`_5$-T_6-$T`_7$-$T_8$-$\underline{T`_9}$-$\underline{T`_{10}}$-$T_{11}$-$T`_{12}$-T_1}

(b)

DCS={...-(-4)(E_1)-...-(4)(E_2)-...} Equation 5.19

JS={T_1-$T`_2$-$T_3$-$\underline{T`_4}$-$\underline{T`_5}$-$T_6$-$T`_7$-T_8-$T`_9$-$\underline{T_{10}}$-$\underline{T_{11}}$-$T`_{12}$-T_1}

where T, $T`$ and underlined junctions are the direct, corresponding and repeated junctions.

As explained, ±4 in the DCS are the difference codes for end points. The end points and junctions inside the tracing contour can be detected by the DCS. Also all the Junctions and their sequences have been found by windowing and applying the junction detection algorithm. Each underlined pair junction in the JS of equation 5.18 and 5.19 shows that a junction has been visited twice during tracing therefore a discontinuity has occurred in the loops. The detection of repeated junctions can be done easily by checking the junctions with the same co-ordinate in the T_L array. For each end point in the DCS, a pair of repeated junctions are available in JS.

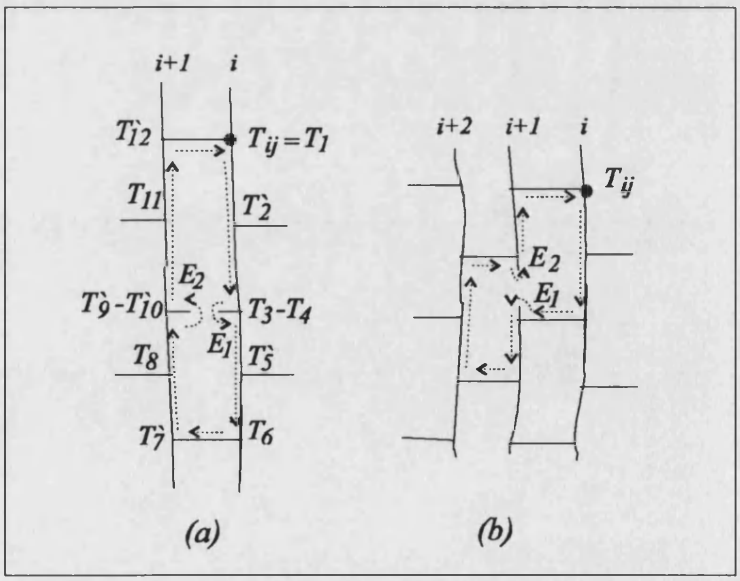


Figure 5.33: Loop tracing inside the broken loops when a discontinuity is on the (a) horizontal or (b) vertical line, all the junctions are available. The start point for tracing is always the direct junction on the line (T_{ij}).

From DCS, the number and direction of the end points are available n_e=2, for condition (a) and (b) the end points are: [E_1(LEP), E_2(REP)] and [E_1(TEP), E_2(BEP)]. From JS the number of junctions for the disconnected loops is n_t=10 which shows that two loops with all the junctions are available in the disconnected area. The correct sequence in JS, direct and corresponding junctions consecutively repeated (...-T-$T\hat{}$-T-$T\hat{}$-...), must be obtained when all the junctions on the border loops are available. Because one of the repeated junctions is a direct junction and the other is a corresponding junction, the problem is related to the only one discontinuity on the lines, so the end points (E_1 to E_2) are connected by applying the line filling algorithm.

The information of the above conditions is summarised in Table 5.1. The number of end points and junctions is defined as n_e and n_t.

End points (n_e=2)	Repeated junctions for n_t=10	Discontinuity on line
(a) E_1(LEP)-E_2(REP)	(a) Direct-Corresponding	(a) Horizontal
(b) E_1(TEP)-E_2(BEP)	(b) Corresponding-Direct	(b) Vertical

Table 5.1: The information of the loops when only one discontinuity occurs on a line

b) Double discontinuities on lines

Sometimes double or multiple discontinuities may occur for the lines. Figure 5.34 shows examples of double discontinuities on the lines when all the junctions are available inside the broken loops.

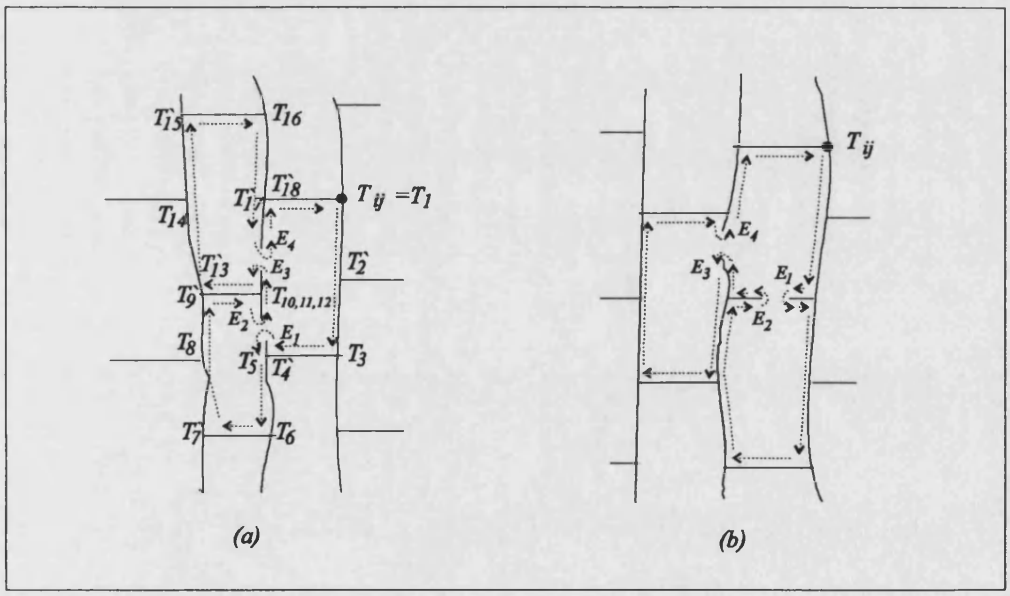


Figure 5.34: Multiple discontinuities on the lines so that a junction is repeated three times in JS.

The difference code and junction sequences are written here,

(a) Equation 5.20

DCS={...-(-4)(E_1)-...-(4)(E_2)-...-(-4)(E_3)-... -(4)(E_4)-...}

JS={T_1-$T\hat{}_2$-T_3-$\underline{T\hat{}_4}$-$\underline{T\hat{}_5}$-T_6-$T\hat{}_7$-T_8-$\underline{T\hat{}_9}$-$\underline{T_{10}}$-$\underline{T_{11}}$-$\underline{T_{12}}$-$\underline{T\hat{}_{13}}$-T_{14}-$T\hat{}_{15}$-T_{16}-$\underline{T\hat{}_{17}}$-$\underline{T\hat{}_{18}}$-T_1}

(b) Equation 5.21

DCS={...-(4)(E_1)-...-(-4)(E_2)-...-(4)(E_3)-...-(-4)(E_4)-...}
JS={T_1-T'_2-$\underline{T_3}$-$\underline{T_4}$-T'_5-T_6-T'_7-$\underline{T_8}$-$\underline{T'_9}$-$\underline{T'_{10}}$-$\underline{T'_{11}}$-$\underline{T_{12}}$-T'_{13}-T_{14}-T'_{15}-$\underline{T_{16}}$-$\underline{T_{17}}$-T'_{18}-T_1}

The number and direction of the junctions and end points are classified in Table 5.2. By considering the DCS, four end points are available in the disconnected area which provide two pair points for each disconnected line. Four repeated junctions in the JS confirm the number of end points. The total number of junctions and the correct sequences in the JS show that all the junctions are available but that a junction is repeated more than twice in the junction sequence, therefore a double discontinuity has occurred on the lines. The algorithm connects the pairs of the end points, which are positioned in the sequence consecutively, (E_1, E_2) and (E_3, E_4), using the filling algorithm.

End points (n_e=4)	Repeated junctions for n_t=13	Discontinuity on line
(a) E_1(TEP)-E_2(BEP)-E_3(TEP)-E_4(BEP) (b) E_1(LEP)-E_2(REP)-E_3(TEP)-E_4(BEP)	(a) Three corresponding (T'_4-T'_5, T'_9-T'_{13} and T'_{17}-T'_{18}) and one direct (T_{10}, T_{11} and T_{12}) (b) Three direct (T_3-T_4, T_8-T_{12} and T_{16}-T_{17}) and one corresponding (T'_9, T'_{10} and T'_{11})	(a) Vertical (b) Horizontal-Vertical

Table 5.2: The information of the loops when double discontinuities occur on lines

5.1.4.5 Discontinuity on a junction

The loop tracing now is tested on a disconnected area with one junction missing as shown in Figure 5.35. The proposed method by generating the DCS and JS is applied on the area from the start pixel.

Figure 5.35 shows that three loops are involved for any discontinuity on a junction, the difference code and junction sequences are,

(a): Equation 5.22
DCS={...-(4)(E_1)-...-(-4)(E_2)-...-(4)(E_3)-...}
JS={T_1-T'_2-$\underline{T_3}$-$\underline{T_4}$-T'_5-T_6-T'_7-$\underline{T_8}$-$\underline{T_9}$-T'_{10}-T_{11}-T'_{12}-$\underline{T_{13}}$-$\underline{T_{14}}$-T'_{15}-T_1}

(b): Equation 5.23
DCS={...-(-4)(E_1)-...-(-4)(E_2)-...-(4)(E_3)-...}
JS={T_1-T'_2-T_3-$\underline{T'_4}$-$\underline{T'_5}$-T_6-T'_7-T_8-$\underline{T'_9}$-$\underline{T'_{10}}$-T_{11}-T'_{12}-T_{13}-$\underline{T'_{14}}$-$\underline{T'_{15}}$-T_1}

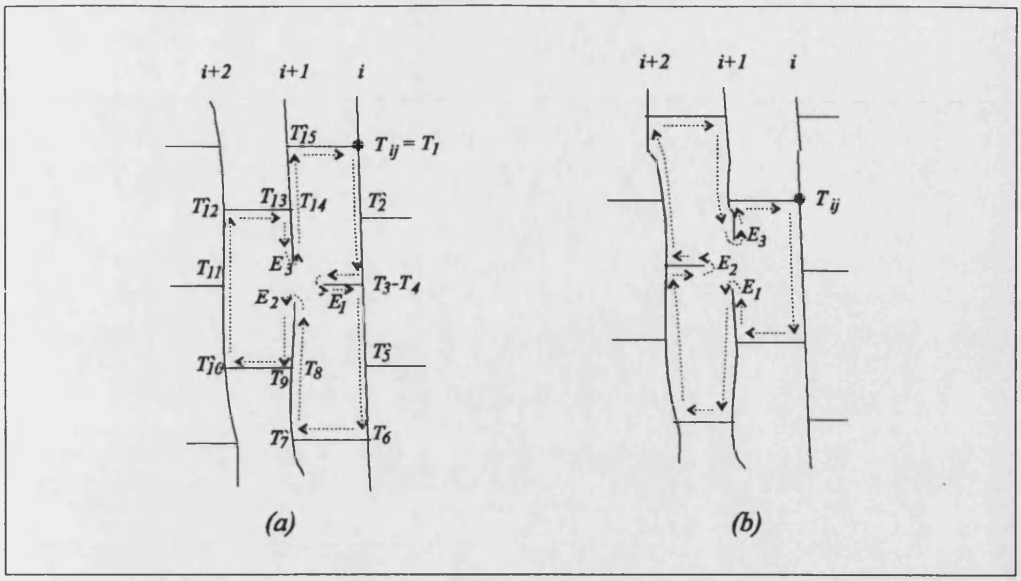


Figure 5.35: A disconnected junction among three loops, (a) corresponding junction and (b) direct junction.

The information of the disconnected area is summarised in Table 5.3. The number of the end points (n_e=3) and junctions (n_t=12) which determine that three disconnected pixels and twelve junctions are available in the loops. By considering the sequence of the repeated junctions and direction of the end points, the missing junction can be predicted. For condition (a), three direct junctions are repeated in the sequence, therefore a corresponding junction is surrounded by these three junctions. But the missing junction in (b) is a direct junction among the three corresponding junctions. Also the direction of the end points in the DCS prepares the information for reconstructing a junction.

After classification of the missing junction (number, type and position) by considering the repeated junctions and end points (E_1, E_2, E_3), the algorithm can rebuild the missing junction and its connected lines to the end points by applying the filling algorithm. The end points of the vertical line are connected first, then the cross point of the filled line with a horizontal line from the other end point is indexed as the missing junction. For condition (a), the line between E_2 and E_3 make a vertical line and the cross point of this line and the horizontal line from E_1 ($y=y_{E1}$) is the missing junction point. For the condition (b), the first (E_1) and third (E_3) end points are connected together and then a horizontal line from the second end point with its y co-ordinate ($y=y_{E2}$) will be connected to the filling line.

End points (n_e=3)	Repeated junctions for n_t=12	Missing Junction
(a) E_1(LEP)-E_2(TEP)-E_3(BEP) (b) E_1(TEP)-E_2(REP)-E_3(BEP)	(a) Three direct (T_3-T_4, T_8-T_9 and T_{13}-T_{14}) (b) Three corresponding (T_4-T_5, T_9-T_{10} and T_{14}-T_{15})	(a) Corresponding (b) Direct

Table 5.3: The information of the loops when discontinuity occurs on a junction

5.1.4.6 Discontinuity for two neighbour junctions (direct and corresponding)

One step further is to consider a discontinuity for two adjacent junctions. In a large area of discontinuity usually two relative junctions, direct and corresponding, both are missing as shown in Figure 5.36. With this condition, no direct information from the lost junctions is available and the method for correcting the loops is interpretation of the junctions by considering the nearest neighbouring junctions. Now four loops are involved inside the disconnected area.

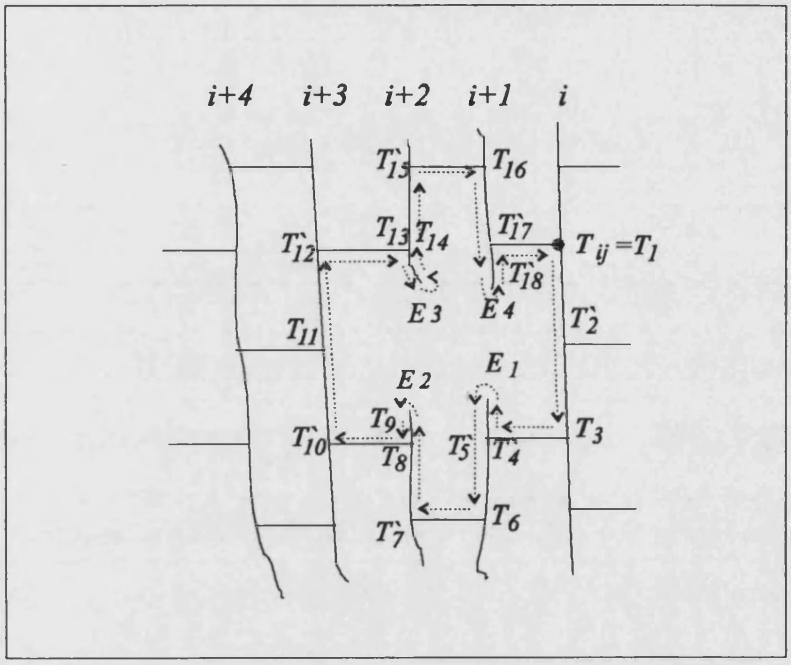


Figure 5.36: Missing two junctions of a loop

The tracing algorithm scans inside the closed contour using the inverse directional coding. The DCS and JS are written in equation 5.24,

DCS={...-(-4)(E_1)-...-(-4)(E_2)-... -(4)(E_3)-...-(4)(E_4)-...} Equation 5.24

JS={T_1-$T`_2$-$T_3$-$\underline{T`_4}$-$\underline{T`_5}$-$T_6$-$T`_7$-$\underline{T_8}$-$\underline{T_9}$-$T`_{10}$-$T_{11}$-$T`_{12}$-$\underline{T_{13}}$-$\underline{T_{14}}$-$T`_{15}$-$T_{16}$-$\underline{T`_{17}}$-$\underline{T`_{18}}$-T_1}

The prediction of missing components can be done by processing the above statements. The number of end points (n_e=4) and their directions from DCS show that two vertical lines are disconnected inside the loops. The number of junctions (n_t=14) also shows that four loops are involved for this type of discontinuity but two junctions are missing. The number of the repeated junctions is equal to the number of end points where each pair of the end points has been made by a missing junction. The information of this type of discontinuity is summarised in Table 5.4.

End points (n_e=4)	Repeated junctions for n_t=14	Missing Junctions
-E_1(TEP), E_4(BEP)	-Corresponding ($T`_4$-$T`_5$ and $T`_{17}$-$T`_{18}$)	-Direct
-E_2(TEP), E_3(BEP)	-Direct (T_8-T_9 and T_{13}-T_{14})	-Corresponding

Table 5.4: The information of the loops when discontinuity occurs on two junctions

To find an accurate point for the missing junction on the filled vertical line has been studied. *Chia et. al* [110] proposed to make use of the geometric property of cross ratio to improve the accuracy of grid junction locations in grid-coded images through a statistical error analysis. Although with their method we can calculate the position of the missing junction, the results are acceptable only for a smooth and monotonous objects. For the facial images, with unpredictable deformities and shapes, the extrapolation or statistical methods are not acceptable therefore the best approach is to use the information from the neighbouring loops and corresponding junctions adjacent to the missing loop.

The junction sequence is a good place to find the whole information about the loop. The junctions on the first scanning vertical line (i) has been used to interpret the y co-ordinate of the missing junction. For interpreting the junctions when the first line is the reference line (i=0), the median point between the two direct junctions is calculated as the relative junction. The interpretation of missing junctions by using the relative information from other neighbour junctions is applied on the disconnected areas and a reasonable result has been achieved.

For correcting the junctions two approach are available, (1) correcting the whole area of the discontinuity in one path, and (2) reducing the problem area by correcting the loops on the first vertical line and then line by line. Both methods are explained here.

(1): By considering the number and direction of end points and also their sequences (E_1, E_2, E_3, E_4), it is clear that the first and last end points (E_1-E_4) and the second and third end points (E_2-E_3) make the pair end points. Each pair belongs to a vertical line and therefore can be connected together. The missing junctions are positioned on the filled lines and must be interpreted from the traced loops.

From the junction sequence, the y co-ordinate of the junction T_2 (relative junction) is used to recover the junctions between the two filled lines. The cross points of the vertical filled lines with the y co-ordinate of the relative junction (y=y_{T2}) are indexed as the missing junction points.

(2): The algorithm needs only to determine the end points for the first vertical line, the first BEP and the last TEP. Then the gap between these two points is completed by applying the filling algorithm. The tracing algorithm re-starts from the start pixel of the loop. To have a complete loop on the first line , the position of the missing junctions (median junction for the loops on the first line) in the retraced loops is obtained by using the relative junction information (y=y_{T2}). The reduced problem area on the next line, as shown in Figure 5.37, is the same condition as explained before in section 5.1.4.5 (a) and therefore can be corrected easily when the loops on subsequent lines are scanned.

The advantage of the first method is the fast processing of the total discontinuity area only in one pass. But it is found that using the second method is more robust, because of tracing the discontinuity area several times and correcting the loops line by line and this is the method employed.

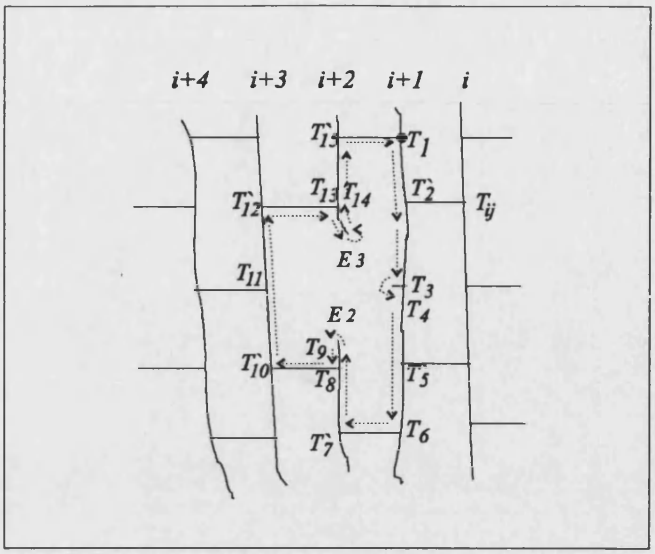


Figure 5.37: Correcting the discontinuities line by line and reducing the problem area

5.1.4.7 A large area of discontinuities

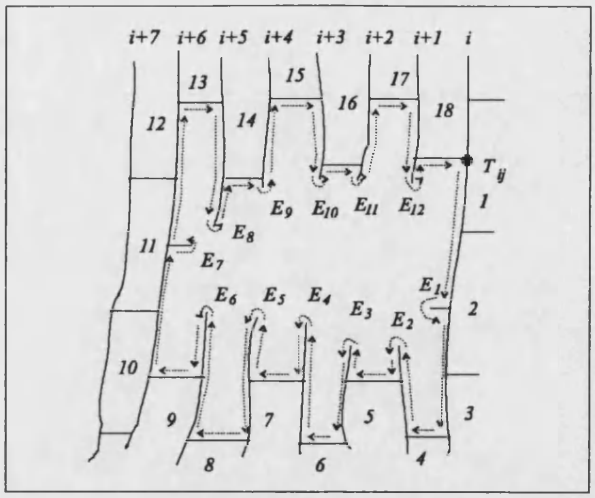
The simple but basic problems on the loops and the novel methodology for classifying and solving them by using the defined arrays and sequences have been described. Now a large area of discontinuities can be tested by the proposed method and then a general algorithm will be used. The problems caused by overlapping are explained later (5.1.4.8). An example of a large discontinuity area is shown in Figure 5.38 (a).

The tracing algorithm scans inside the broken loops by inverse directional coding from T_{ij}. All the information from the loops are stored in E_L, T_L and DCS which will be used for classifying the loops components. The junction sequence (JS) only is written here,

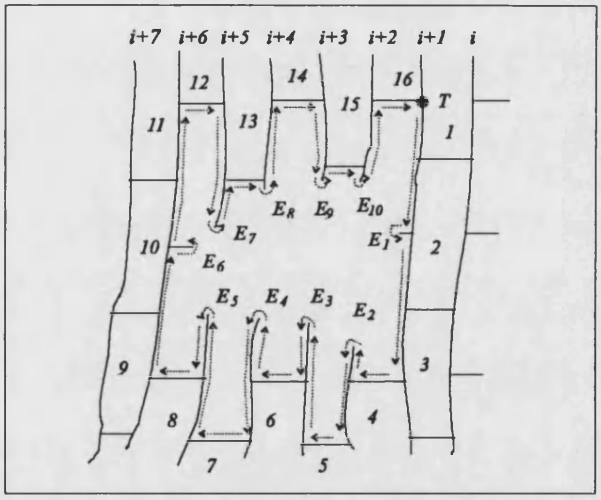
JS={T_1-$T`_2$-$\underline{T_3}$-$\underline{T_4}$-$T`_5$-T_6-$T`_7$-$\underline{T_8}$-$\underline{T_9}$-$\underline{T`_{10}}$-$\underline{T`_{11}}$-$T_{12}$-$T`_{13}$-$\underline{T_{14}}$-$\underline{T_{15}}$-$\underline{T`_{16}}$-$\underline{T`_{17}}$-T_{18}-$T`_{19}$-$\underline{T_{20}}$-$\underline{T_{21}}$-$T`_{22}$-T_{23}-$\underline{T`_{24}}$-$\underline{T`_{25}}$-T_{26}-$T`_{27}$-$T_{28}$-$\underline{T`_{29}}$-$\underline{T`_{30}}$-$\underline{T_{31}}$-$\underline{T_{32}}$-$T`_{33}$-T_{34}-$\underline{T`_{35}}$-$\underline{T`_{36}}$-$\underline{T_{37}}$-$\underline{T_{38}}$-$T`_{39}$-$T_{40}$-$\underline{T`_{41}}$-$\underline{T`_{42}}$-T_1}

Equation 5.25

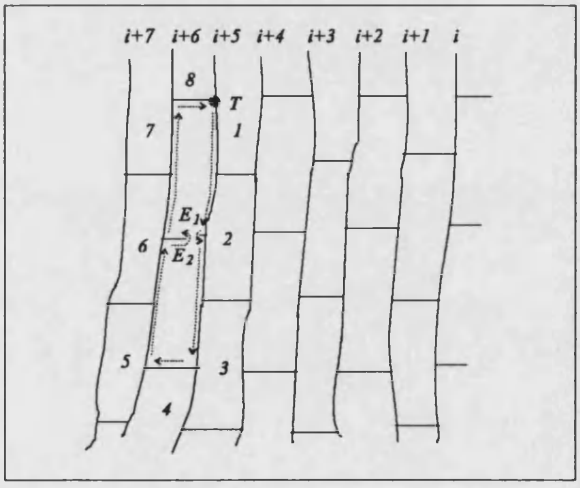
From the number of end points (n_e=12) and number of junction (n_t=30), it is clear that a large area of discontinuity is available. The junction sequences in JS are correct (*T*-*T`*-*T*-*T`*-...) and also the number of the repeated junctions are equal to the number of the end points therefore there is no overlapping problem. Classifying the end points as a pair group can be done by considering the close contour and direction of the end points (TEP-BEP) in the sequence (E_2-E_{12}, E_3-E_{11}, E_4-E_{10}, E_5-E_9, E_6-E_8).



(a)



(b)



(c)

Figure 5.38: (a) A large area of discontinuities, the boundary loops are 1-18 and end points (E_1-E_{12}). (b) The disconnected loops on the line *i* are corrected and the tracing algorithm starts re-tracing from T_{ij}. (c) When all the loops on the line *i* are corrected, the loop tracing starts from the start pixel of the line (*i+1*). The part of the disconnected area is detected and then the loops on this line are corrected. This process is continued for all the lines and the broken loops on each line will be corrected in an orderly manner.

As explained, to achieve a robust and accurate method for correcting the discontinuity area, the loops are corrected line by line. The algorithm corrects only the loops on the first line (i) and then the tracing process will carry out from the start junction again (T_{ij}). The algorithm fills the gaps between the pair end points for the first line (E_2 and E_{12}) and then a horizontal line with the y co-ordinate of the relative junction ($y=y_{E1}$) is drawn from the point E_1 to make a junction as shown in Figure 5.38 (b).

When all the loops on the line (i) are traced and corrected, the next line ($i+1$) is indexed for tracing and the part of this disconnected area, covered by line $i+1$, will then be corrected. The process is continued until correcting the last loop on the line ($i+5$). The procedure for correcting the disconnected area by using the explained method is shown in Figure 5.38 (b, c, and d).

The loop tracing process for recovering any discontinuity is summarised here,

1. Apply the inverse directional loop tracing from the start pixel (T_{ij}).

2. Generate the loop network (E_L and T_L) along with the difference code and junction sequences (DCS and JS).

3. Classify the end points, number (n_e), direction (e_d) and pair groups.

4. Find the number of junctions (n_t) and check their sequences (T-T^-T-T^-...) for determining the number of involved loops and problem area. Table 5.5 provides a summary of the basic discontinuities explained previously.

Condition	n_e	n_t	Involved loops
One discontinuity on a line	2	10	2
Double discontinuities on lines	4	13	3
A missing junction	3	12	3
Missing two junctions	4	14	4

Table 5.5: Some information from the general discontinuities

5. Find the repeated junctions to confirm the number of end points, as predicted in stage 3, and type of the discontinuity.

6. Fill the vertical gap between the first pair of the end points, The first TEP and the last BEP, on the first disconnected line by applying the filling algorithm.

7. Determine the missing junctions, direct and corresponding, on the filled line by considering the LEPs and the relative junctions in the sequence. If a LEP is available on the previous line, it will be extended by drawing a horizontal line with the equation of $y=y_{LEP}$ until crossing the filled line. If there is not any end point for identifying the position of the missing junctions, the relative junctions on the traced line, from the JS, will be used for indexing the junctions. If the traced line is the reference line without any median junction, the median of the two direct consecutive junctions will be calculated and indexed as the relative junctions.

8. Go back to stage 1 and start tracing again from T_{ij}.

The process for other remaining lines is repeated and the gaps will be corrected in an orderly manner.

5.1.4.8 Discontinuity along with overlapping

The sharp area on the nose usually generates overlapping of the projected lines. The last step is to evaluate the proposed method on the overlapped area and then find a solution for covering it. Consider that an overlapping occurs on the lines $i+1$ and $i+2$, as shown in Figure 5.39, along with discontinuity and missing a junction.

Figure 5.39 (a) shows that the overlapped line is disconnected and two end points are available for this line (E_2 and E_3). The sequences of this condition are written in equation 5.9. Three end points and their directions (E_1(LEP), E_2(TEP) and E_3(BEP)) show a missing junction, but only two repeated junctions are available in the JS. A cross point of two lines are detected during the loop tracing and junction detection algorithm. This point (T_o) is indexed by an attribute number (a_t=4, the number of lines connected to the candidate pixel) in the junction array to distinguish it from the junctions. The incorrect sequence in the JS, ...-T_8-T_9-..., is an indication for the overlapping problem. Therefore the overlapping point is the point T_o between the two incorrect junction sequence. The missing junction can be reconstructed as explained in section 5.1.4.5.

DCS={...-(4)(E_1)-...-(-4)(E_2)-...-(4)(E_3)-...}
JS={T_1-$T`_2$-$\underline{T_3}$-$\underline{T_4}$-$T`_5$-T_6-$T`_7$-$T_8$-($T_o$)-$T_9$-$T`_{10}$-$\underline{T_{11}}$-$\underline{T_{12}}$-$T`_{13}$-T_1} Equation 5.26

where T_o is detected as the cross point of four lines.

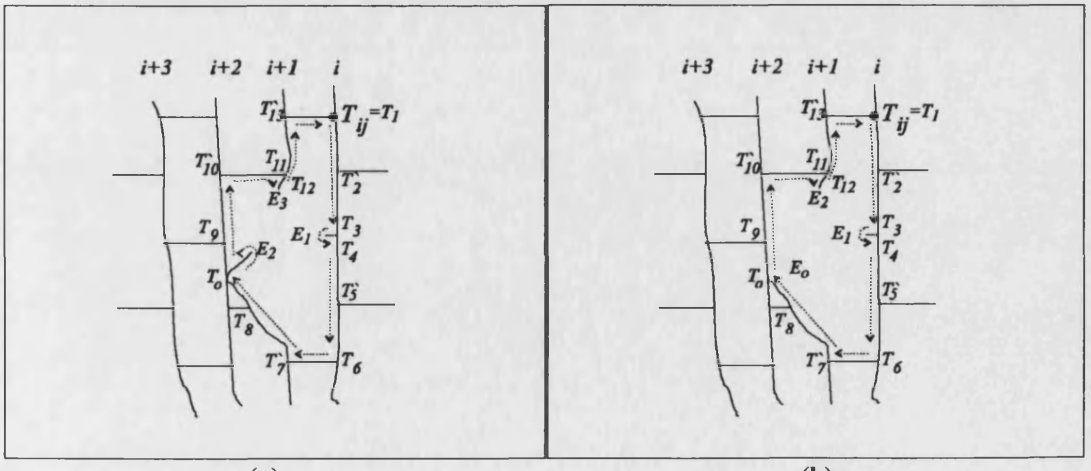


Figure 5.39: Overlapping of two lines $i+1$ and $i+2$, (a) an end point is available for the overlapped line inside the loop, and (b) the end point and overlapped points are the same.

Figure 5.39 (b) shows that the overlapped and end points both are positioned on the next line ($i+2$). It is possible that the overlapping point is detected as a junction point, or it is not. First suppose that the overlapping point is not detected as a junction, but it is detected as a cross point (T_o) of the three lines. An attribute number is assigned for this point (a_t=3) during the tracing and then the point co-ordinate and attribute number are saved in the junction array. For this condition the sequences are written in equation 5.27,

DCS={...-*(4)*(E_1)-...-*(4)*(E_2)-...}
JS={T_1-$T\hat{}_2$-$\underline{T_3}$-$\underline{T_4}$-$T\hat{}_5$-T_6-$T\hat{}_7$-T_8-(T_o)-T_9-$T\hat{}_{10}$-$\underline{T_{11}}$-$\underline{T_{12}}$-$T\hat{}_{13}$-T_1} Equation 5.27

where To is detected as the cross point of three lines.

The number and direction of the end points (E_1(LEP) and E_2(BEP)) is not enough for correcting a missing junction. But from the number of junctions (n_t=11) and information of Table 5.5, it is found that a junction is not available in the sequence. The incorrect sequence in the JS (...-T_8-T_9-...) also indicates the overlapping problem in the loops. The overlapping point is positioned between the two incorrect sequence and therefore the detected cross point (T_o) must be the overlapping point. Correcting the discontinuity area can be done by considering the overlapping point as the missing BEP and following the proposed method in section 5.1.4.5.

The next stage is to consider an overlapping point as a junction. When the overlapped line is detected as a junction, the sequence of the junctions may be correct or not. The overlapping problem mostly generates a corresponding junction for the left side of a face and a direct junction for the right side because of the facial curvature. The junction sequence for Figure 5.39 (b), when the overlapping point is a junction, is written in equation 5.28.

JS={T_1-$T\hat{}_2$-$\underline{T_3}$-$\underline{T_4}$-$T\hat{}_5$-T_6-$T\hat{}_7$-T_8-$T\hat{}_o$-T_9-$T\hat{}_{10}$-$\underline{T_{11}}$-$\underline{T_{12}}$-$T\hat{}_{13}$-T_1} Equation 5.28

where T_o is detected as a corresponding junction.

Although the overlapping problem is available in the loops but the sequence of the junctions is correct (...-T_8-$T\hat{}_o$-T_9-...). This is because the overlapped point is detected as a junction and it may occur in any place on the next line. To solve this problem, the number of junctions and their sequences is compared with the basic discontinuities, Table 5.5.

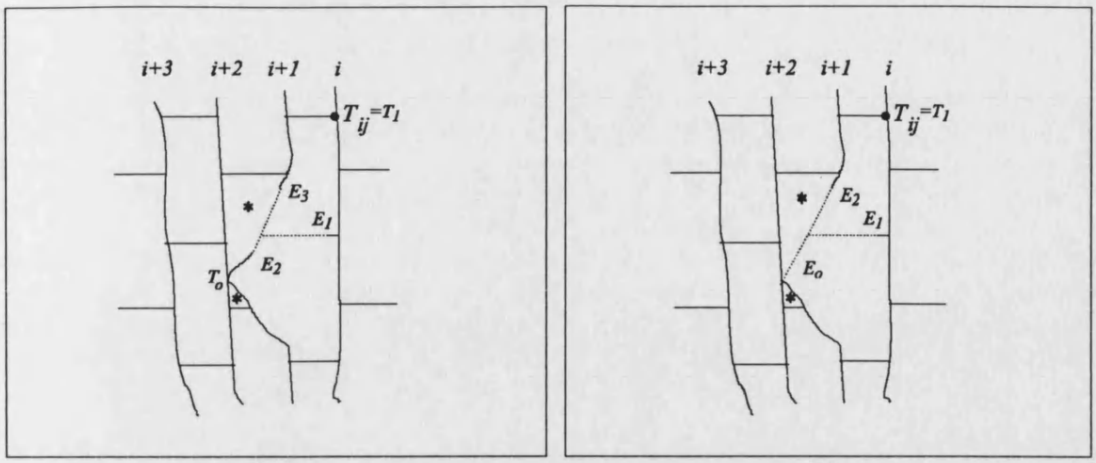


Figure 5.40: The problem of overlapping compensated during the tracing and then the filling algorithm corrected the lines. The exceptional loops remain on the lines as shown by (*).

From the number of junctions (n_t=12) and direction of the end points, the problem is caused by missing a junction. By comparing of the above equation with equation 5.22, $\hat{T}_o$ is positioned incorrectly and therefore must be the overlapping point for the missing end point. The overlapping point is labelled with E_o and added to the E_L array in the correct place. When the overlapping point is determined, correcting the loops can be done using the filling algorithm. The corrected loops for the overlapping conditions of Figure 5.39 are shown in Figure 5.40.

It must be mentioned that the overlapping problem usually generates some exceptional loops without all the necessary junctions. The exceptional junctions are saved to illuminate them during the tracing algorithm. Figure 5.40 shows the corrected loops after applying the filling algorithm. The uncompleted loops, shown by *, are indexed on the line ($i+1$), but the line itself is complete without any discontinuities.

When an overlapping point is determined as a junction, the only way to identify it is by processing the sequences using the basic discontinuities as classified in Table 5.5. Figure 5.41 shows the overlapping of the line $i+1$ and on the line $i+2$ when the overlapped point (E_o) is detected as a corresponding junction ($\hat{T}_o$).

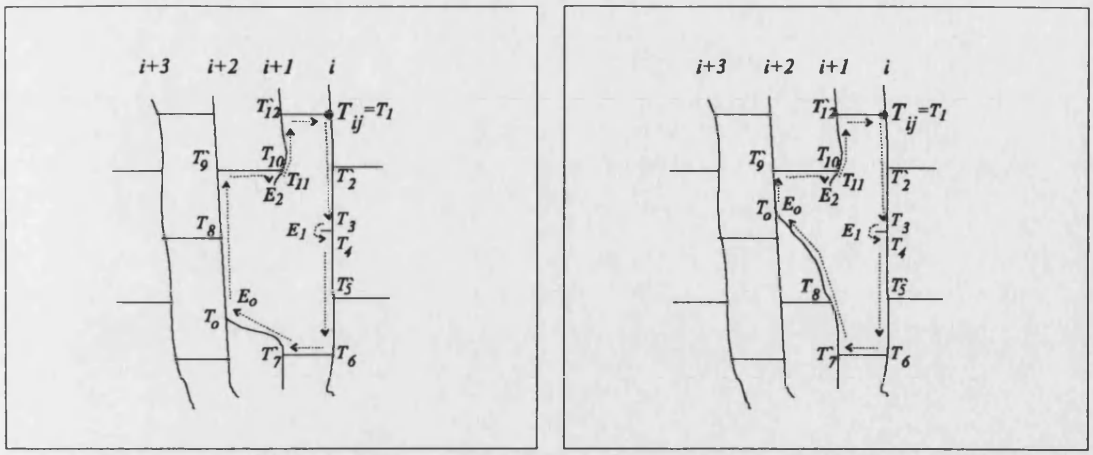


Figure 5.41: The overlapped point is detected as a corresponding junction ($T`_o$) and therefore the junction sequence is not correct, (a) $T`_7$-$T`_o$, and (b) $T`_o$-$T`_9$.

The junction sequences for the both conditions are written in equations 5.29 and 5.30,

(a) JS={T_1-$T`_2$-$\underline{T_3}$-$\underline{T_4}$-$T`_5$-T_6-$T`_7$-$T`_o$-T_8-$T`_9$-$\underline{T_{10}}$-$\underline{T_{11}}$-T_{12}-T_1} Equation 5.29

(b) JS={T_1-$T`_2$-$\underline{T_3}$-$\underline{T_4}$-$T`_5$-T_6-$T`_7$-$T_8$-$T`_o$-$T`_9$-$\underline{T_{10}}$-$\underline{T_{11}}$-T_{12}-T_1} Equation 5.30

where T_o is detected as a corresponding junction.

Two incorrect sequences are available in the junction sequences, (a) $T`_7$-$T_o$ and (b) $T`_o$-$T`_9$, which can be used to solve the problem. By considering the number of junctions (n_t=11) and end points (n_e=2), the condition of missing a junction has occurred for the loops. The position of the overlapped point can be obtained by comparing the junction sequence with the equation 5.22. The overlapped point is the first visited problem in the sequence. For condition (a), the first problem is exactly occurred after $T`_7$ , a direct junction is necessary after $T`_7$, therefore $T`_o$ is the overlapping junction. But the visited problem for condition (b) is occurred after $T_8$, a direct junction is necessary after $T_8$, therefore $T`_o$ is the overlapping junction. The missing end point (E_o) is defined on the overlapped junction (T_o).

The general algorithm for correcting a large area of discontinuity, based on the tracing line by line and reducing the problem area as smaller as possible, is explained in the previous section (5.1.4.7). The overlapping problem can also be detected and corrected if the following stages are considered during the loop tracing.

1. Find the number of end points and their directions to evaluate the problem area.
2. Find the number of junctions and repeated junctions to determine the discontinuity problem by considering the information of the stage 1, then classify the problem area in one of the basic conditions (Table 5.5).
3. Determine the overlapping problem from the incorrect junction sequence and cross points (a_t=3 or 4) in the JS.
4. Determine the overlapped point from the stage 3 and by comparing the sequence of the junctions by the basic discontinuity as found in stage 2.

5. Index the overlapped point as the missing end point.
6. Correct the discontinuity by applying the filling algorithm.
7. Determine the missing junction by considering the y co-ordinate of the relative junction on the first line (from JS).
8. Assign the junctions of the incomplete loops.

Now the discontinuities on the facial images can be corrected by applying the line, boundary and loop tracing algorithms and then the labelled vertical lines and junctions can be used for reconstructing the image.

5.1.4.9 Labelling

After explaining the implemented method for correcting the loops based on the boundary and loop networks and also the difference and junction sequences, we can apply the proposed algorithms on the extracted image. The image 'face' after reference line tracing and boundary tracing is shown in Figure 5.42. The discontinuity areas on the image are high lighted to illustrate common discontinuity areas. The discontinuities from the eyebrow, eye, nose and lip are noticeable.

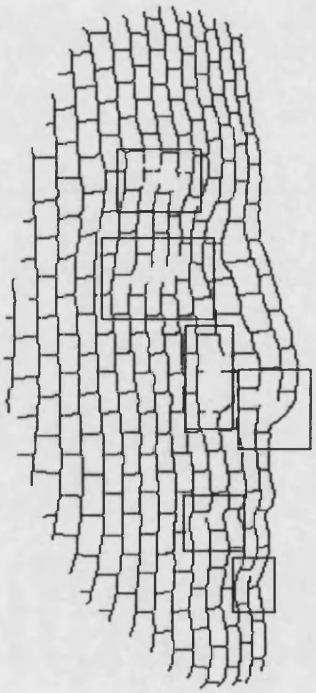


Figure 5.42: The 'face' after reference line tracing and boundary tracing, the discontinuities highlighted only to show the common problem areas on a face.

The loop tracing algorithm starts from the reference line and continues until reaching the last loop on the last line. The process completes the loops on the image based on the information of the loop network and sequences. The 'face' after loop tracing and filling the discontinuity pixels is shown in Figure 5.43.

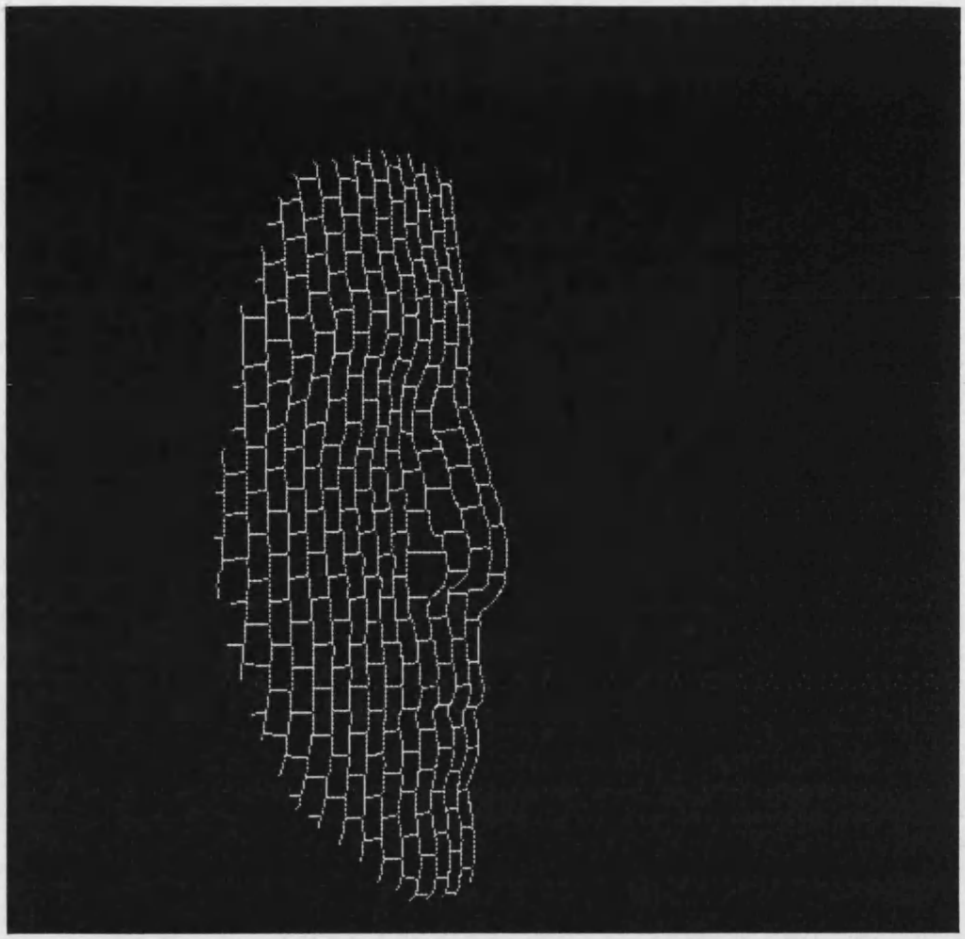


Figure 5.43: Each loop is perfectly formed without any discontinuities

When the loop tracing is completed, all the necessary information about the junctions is available in the JLA (DJLA and CJLA). The labelling process is applied to arrange the junctions in DJLA, CJLA and also exceptional junctions on the boundary network which they have not been visited during the loop tracing. The arrangement of the junctions is done by considering the corresponding slide co-ordinate on the screen and the distance of the junctions from its axes. As the image is perfectly formed without any discontinuities, the vertical lines and their junctions can be separated from the horizontal lines. The tracing on each vertical line by considering the junction labelled arrays give all the labelled information for reconstruction.

The corresponding grid axes matched on the image are illustrated by $X`g$ and $Y`g$ as shown in Figure 5.44. The point (O) is positioned on the centre of the image screen, from the patient positioning, so it is an appropriate start point for arrangement of the junctions on each vertical line. Let each projected junction be indexed by a pair of integers (x_s, y_s), which are the co-ordinates of that junction relative to the screen co-ordinates, and (p_s, q_s), a pair of corresponding slide co-ordinates axes on the screen $X`_g$ and $Y`_g$. p_s and q_s show that the junction belongs to which vertical and horizontal lines from the reference point O. The co-ordinates of the slide junctions was defined by the grid line number rather than pixels. The junction co-ordinates and line numbers have

been saved during the loop tracing in JLA and also boundary tracing in T_B. The junctions on each line are arranged based on the corresponding junctions on the slide.

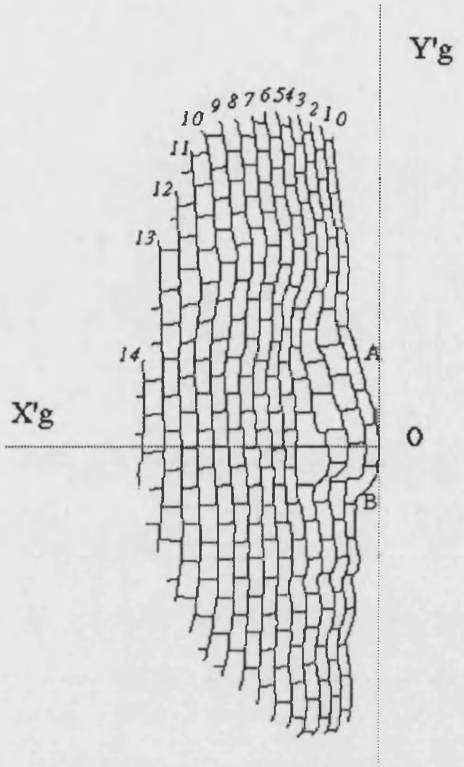


Figure 5.44: The grid axes matched on the traced face to find the correct junction labelling, the vertical lines are numbered from the reference line

Another array has been defined for saving the labelled junctions by considering the junction co-ordinates in JLA and also corresponding grid co-ordinates in the slide as a Labelled Array (LA) so that,

LA={(x_s, y_s, p_s, q_s) where x_s and y_s are the screen co-ordinates for the junctions (JCP), and p_s and q_s are the corresponding junction points based on the slide co-ordinate system matched on the image screen $(X`_g, Y`_g)$}

Equation 5.31

The nearest junction from the $X`g$ axis on each line is the first junction for labelling and the distance of the junctions from this axis gives their orders. The co-ordinates of the junction points on the slide co-ordinate system are defined previously as $d_g(p_g, q_g)$, refer to Figure 5.1. For each junction (p_s, q_s) on the screen co-ordinate system, a corresponding junction on the slide co-ordinate system $d_g(p_g, q_g)$ is available. For example two junctions (A and B) on the reference line are highlighted to show the labelling information as shown in Figure 5.44. The co-ordinates of the point A and B in LA are;

The co-ordinates of point (A) in LA is (x_A, y_A, *0, 3*)
The co-ordinates of point (B) in LA is (x_B, y_B, *0, -2*)

where (x_A, y_A) and (x_B, y_B) are the screen co-ordinates of the junctions. The third number (p_s=*0*) is the number of the vertical line and the fourth number (q_s) prepares the order of the junctions from *X`g* by considering their directions. When each pair of (p_s, q_s) shows a corresponding point on the slide co-ordinate system.

Each junction on the traced image is related to the corresponding junction on the slide. When all the vertical lines scanned and labelled, the image will be ready for reconstruction. Counting and highlighting the lines with a colour which will be assigned to the line position from the reference line will be done simultaneously during the junction indexing. The process has been applied on the lines one by one and highlighted with a range of colours until the last line is achieved. The coloured lines are counted and the total number is saved and will be used for the reconstructed vertical array.

The robustness of the tracking algorithms, tracing and labelling, has been tested by capturing and processing real facial images. Some of the feature extracted and interpreted images from the candidate faces are illustrated in Figure 5.45. The loops on the corrected images are perfectly formed and therefore labelling the junctions and lines can be done readily.

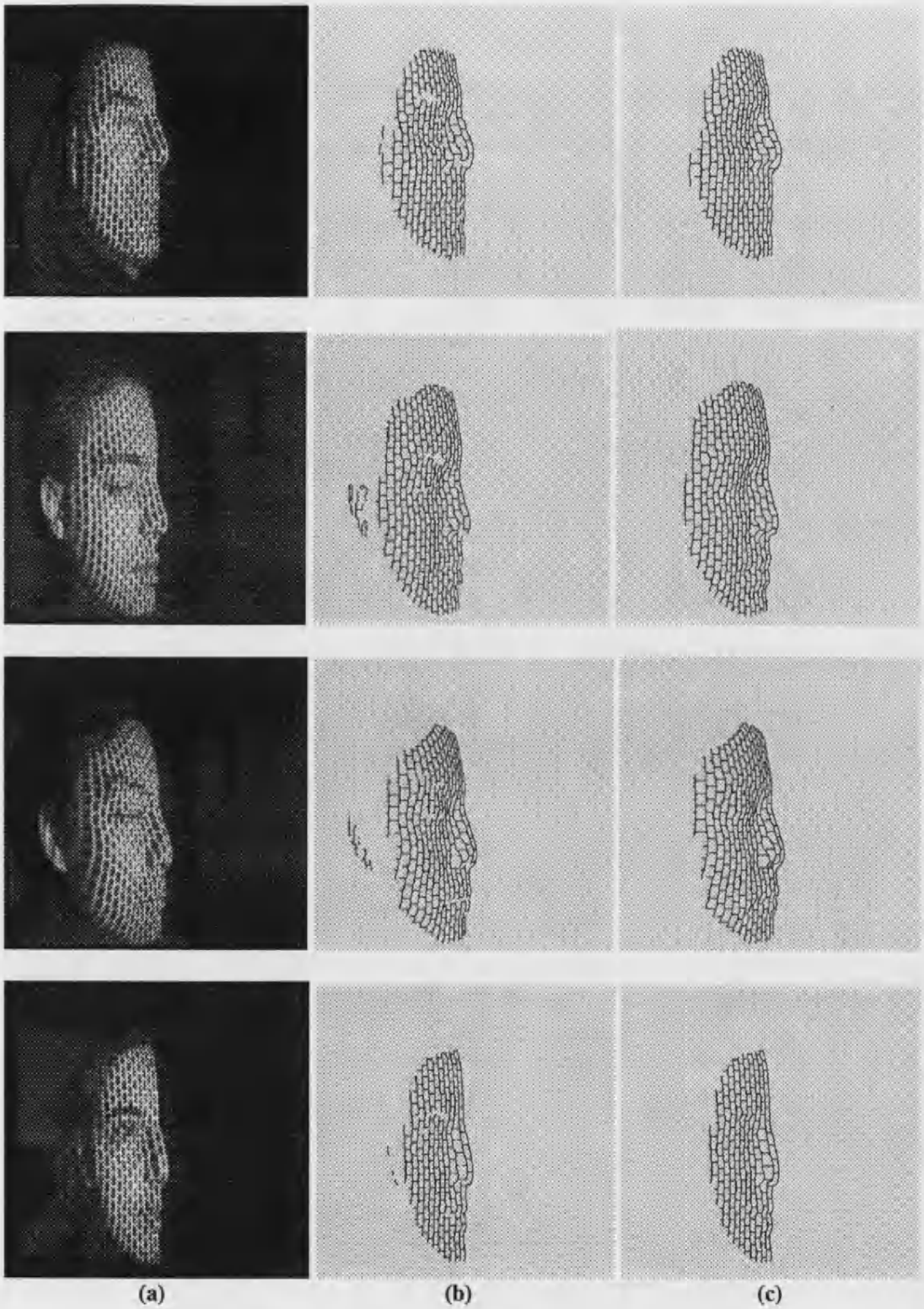


Figure 5.45: (a) Captured images from the left side of the candidate faces, (b) feature extracted images and (c) traced and corrected images.

5.2 Three Dimensional Reconstruction

The image with the traced lines, tagged ends and labelled junctions can now be used to generate a 3-D data array. Each of the junction pixels can be assigned the correct depth or z value. The x and y values of each of these pixels can be taken straight from the co-ordinates of the image. To calculate the 3-D information for each labelled point on the lines, the triangulation method using the calibration information are used. The reconstruction algorithm is done by considering the labelled lines.

For representing a realistic face from the processed image, a 3-D map is generated to store the 3-D information and this is then transferred to the 3-D grid. The first point in the array represents the top-left of a grid. This grid has the line_number equally spaced columns. The total number of assigned pixels on each line for producing the mesh is limited because of the memory management of a PC, therefore a configuration file is considered to allow the user to change the options if the program will not run due to insufficient memory. In this file, the maximum number of lines and maximum vertical resolution on each line should be defined as the necessary information for producing the mesh. The pre-defined line number is 20 and maximum pixels for vertical resolution is 400.

The grid is initially flat by ensuring that all the dimensions in the z-plane are set to zero. The processed image is then used to insert the 3-D information into the grid. This is achieved by relating each of the lines on the image to one of the vertical lines on the grid. Each individual line represents a slice through the subject in a different plane and the array of points can then be plotted to the screen.

After the lines have been height assigned and 3-D array performed, it must be decided how that data is to be presented. There are two approaches:

1. Use the data set to generate an almost photo realistic image in which techniques are used to include skin tone.
2. Use the data set to calculate quantitative information for manipulation by the user.

The first approach is useful in studying the natural history of deformities on a face but having this representation on a computer screen (without using the second approach) does not significantly help in quantifying the deformity. To have a suitable operating environment for processing the reconstructed face, a user friendly and graphical icons environment was designed to generate a reconstructed model of the facial surface.

The capability to draw B-spline curves was incorporated into the reconstruction program [111]. The cubic B-spline has the useful property of using only the nearest three or four samples for any given point on the curve. This makes the B-spline sensitive to adjacent samples unlike a Bezier curve which takes into account all sample points. Bezier and B-spline curves are described in Appendix B.

The implemented wire-frame model as shown in Figure 5.46 is a method for producing a wire mesh around the surface of a face. Although this method shows the contours of the face, it does not produce a solid image of the face and so it is not easy to analyse. Once a set of 3-D points has been generated for the face, a rendering technique is

needed to approximate the surface of the face and then draw a solid representation of the face. The rendering system [112] is produced by using the highest screen mode possible and using an accurate lighting method. The operating environment is a fully functional mouse controller, window style system. this makes the software easy to use by non-experienced users.

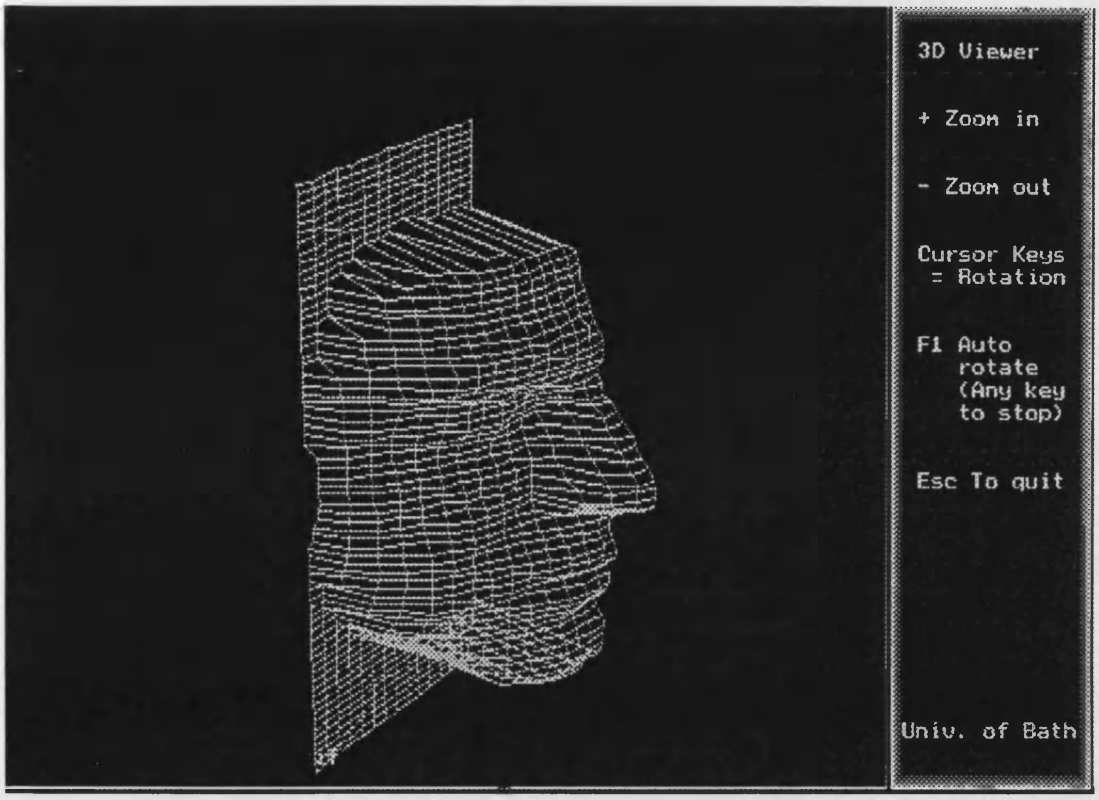


Figure 5.46: Wire-frame reconstructed image for the processed left side of the 'face'.

5.2.1 Representation method

To approximate the surface of the face, from the calculated points, there are three commonly used methods, polygon mesh, parametric cubic curves and the quadric surfaces [113]. For the purpose of this project, using a polygon mesh is the most suitable method because it is readily implemented due to the fact that a list of 3-D points on the face's surface has already been generated and so these points can be used as vertices of the polygon mesh.

A polygon mesh is a collection of vertices, edges and polygons. A vertex is a point that lies on the surfaces of the object to be modelled. An edge connects two vertices and a polygon is the enclosed shape produced by a sequence of edges. A simple polygon mesh is shown in Figure 5.47. The polygons in this example are triangles, but the principle can be applied to polygons with any number of sides.

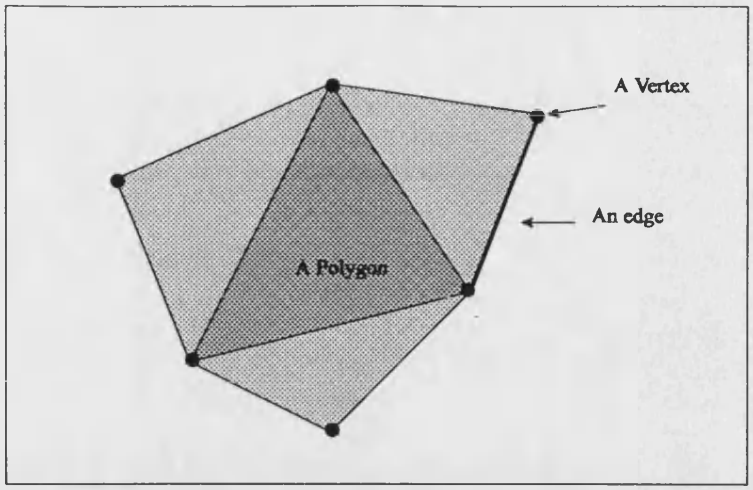


Figure 5.47: A simple polygon mesh

An algorithm was developed for choosing the points from the labelled image as the vertices of the polygon mesh. With this technique, the number of vertices per line of the image varies and so the grid is no longer uniform (an example is shown in Figure 5.48). The algorithm takes each pixel of each line one by one and tries to assign a polygon to it. It does this by looking at the pixel's y co-ordinate, and looking for a pixel with the same y co-ordinate in the next line. If there is not a match, then the algorithm moves to the next vertex. When there is a match, then the current pixel is checked to make sure that it is not the last pixel on the current line and if it is not, then a polygon is added to the polygon list using the three vertices found; the current pixel, the matching pixel on the next line and the next pixel on the current line. The algorithm is repeated for every pixel on all of the lines of the image, except for the last line because there is no line to the right of it and its pixels will have already been used.

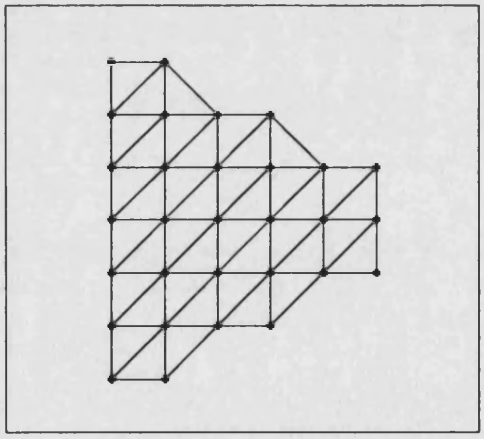


Figure 5.48: A mesh produced by using the new technique

In order to be able to interpolate lighting information across the surface of a polygon, the light intensity must be known for each vertex of the polygon. This means that a surface normal for each vertex of the mesh must be found. The technique for estimating each vertex normal was developed in 1971 by *Gouraud* as part of his Gouraud shading algorithm [114]. The technique finds the normal for each vertex of the polygon mesh by averaging the normals of all of the polygons that use the vertex.

Phong shading is another technique, developed by *Phong* in 1975, which produces a more accurate rendered image. The way in which Phong achieved this improvement was to interpolate the vertex normals across the surface of the polygon, and then for every pixel of the polygon, the light intensity can be found as for Gouraud method. The only drawback that Phong shading has is that a huge number of calculations are required to render the image and so it is very slow.

It is found that to satisfy the project objectives, the face should be Phong shaded to give the best possible image as shown in Figure 5.49. The problem with the Phong shading technique, even if using the faster method, is that on slower computers it can not be drawn real time and so it makes rotating and manipulating the face very slow. To overcome this speed problem, varying levels of rendering can be used, such that when the face is being rotated it is rendered with flat shading, to give an idea of the lighting, and when the user has finished manipulating the face, it is rendered with Phong shading. The style used when the face is being manipulated is known as the 'preview mode' and the style used to draw the final image is called the 'render mode'. In order to take into account that computers are becoming faster and faster, and so may be capable of rendering Fast Phong shading in real-time, the software also allows the user to use Fast Phong shading as the 'preview mode' when manipulating the face.

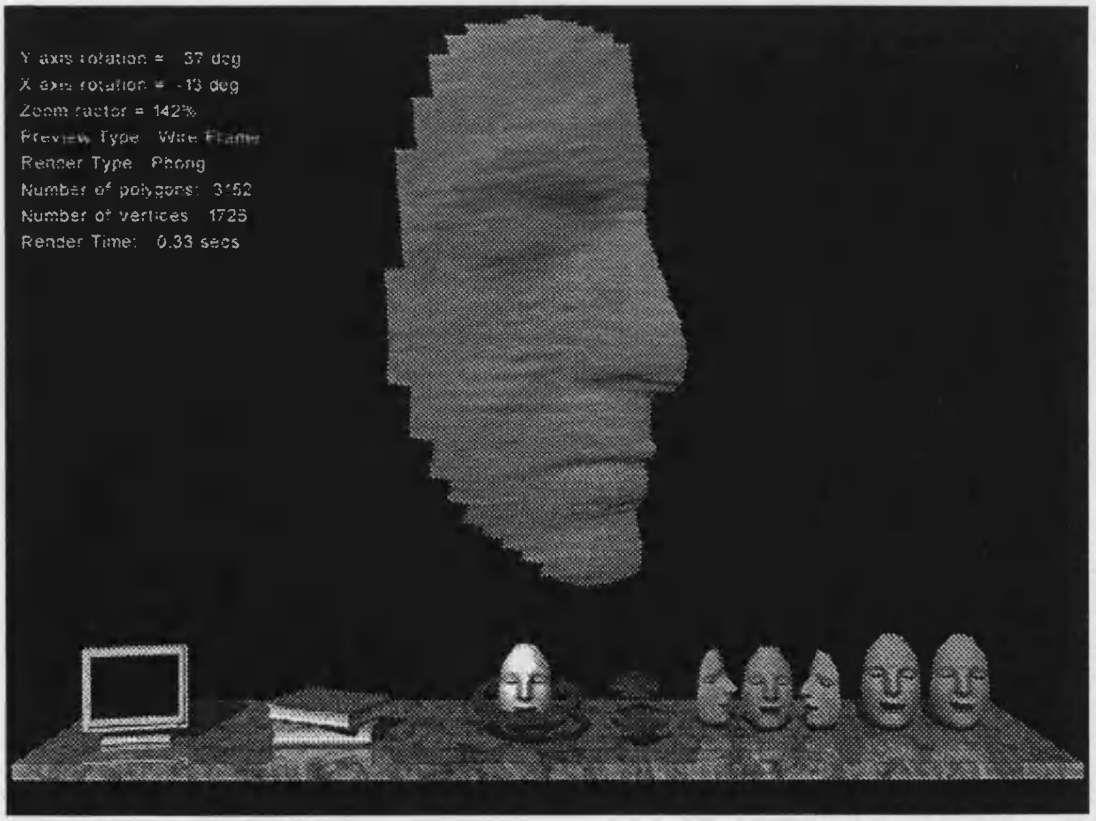


Figure 5.49: The reconstructed image from the face using Phong shading technique

Since the resolution of the polygon mesh has been greatly improved, the problem of quantisation noise has arisen [115]. The result of this problem is that the surface of the reconstructed face contains visible 'ripples'. The noise can be removed by looking at its properties compared to the actual face. The noise can be considered random but the

contours of the face are not random because the displacement of each point on the surface of the face is related to the displacement of the points adjacent to it.

The noise in the system occurs along the length of each distorted lines from the captured image, and so these are the elements that require filtering. Considering the values in each line to be in an array n entries long, then the magnitude for a point i in that array after applying the averaging filter is given below,

$$\textit{Filtered magnitude of point } i = \textit{(Magnitude of point (i-1) + 6} \times \textit{Magnitude of point (i) + Magnitude of point (i+1)) /8} \qquad \text{Equation 5.32}$$

It has been seen that the apparent quantisation noise can be removed by using an averaging filter, but the problem with this kind of filter is that it may also remove very low magnitude, but still significant, information about the surface of the face. An example of this problem is that if the subject had a scar on their face, then it could appear on the polygon mesh as being very similar to a ripple caused by the noise. If the filtering algorithm is then applied to the mesh, the magnitude of the scar may be reduced. For this reason, the software will not automatically filter all meshes produced, but will allow the user to decide whether the mesh should be smoothed or not. Figure 5.50 shows the reconstructed image after applying the smoothed filter.

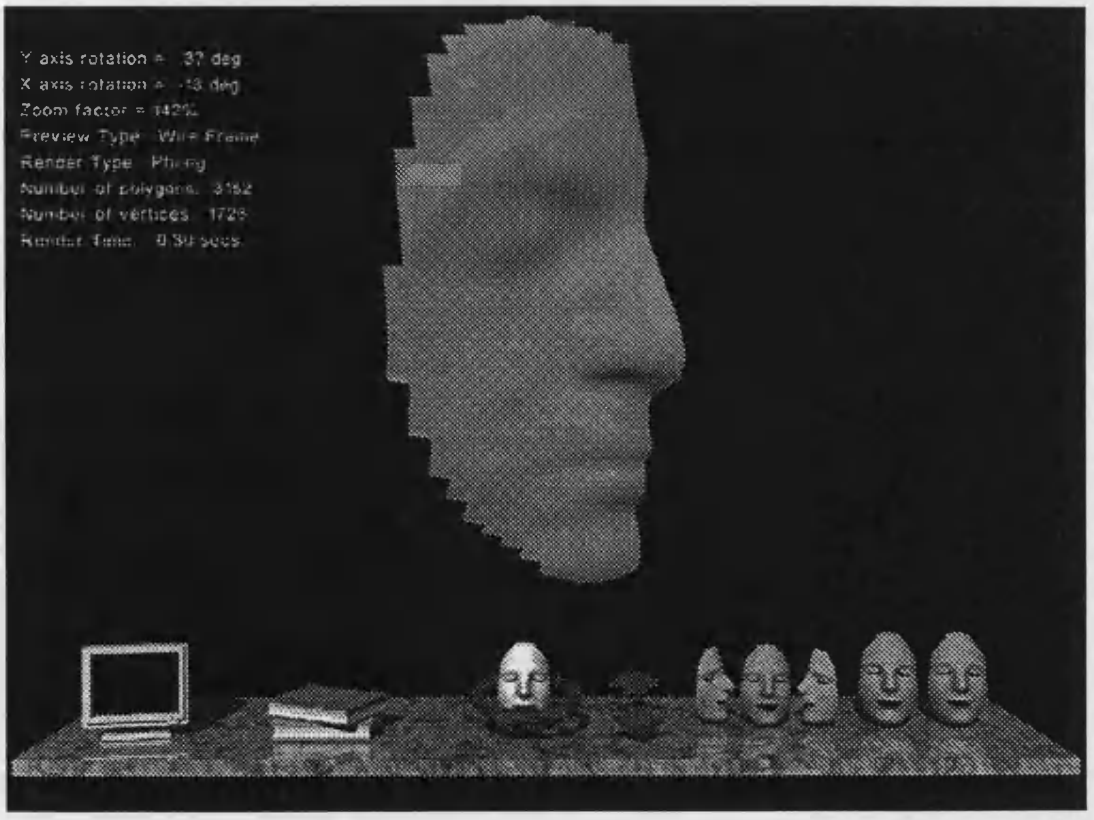


Figure 5.50: Reconstruction of the processed face after applying the smoothed filter, the user friendly environment with graphical icons makes the software easy to use by non-experienced users.

5.2.2 Joining two sides of the face

Each side of a face must be reconstructed separately and so an algorithm has been developed to join the sides for a complete representation of the face. The polygon mesh for each side of the face can be joined by matching the vertices in each mesh produced from the line on each image that is common to both images as shown in Figure 5.51.

The common line will not be exactly the same position in the left and right meshes, due to the fact that it is very difficult to set up the capturing equipment to exactly the same position for each side of the face. For this reason a matching technique is required to adjust one of the meshes in order that it fits correctly with the other mesh. Two types of the matching techniques were tested for joining the sides of a face; slide matching and feature matching.

A slide matching technique matches the positions of the vertices of the left and right representations of the common line. The algorithm then attempts to match the position of the vertices of both meshes by adding an offset to the position of one mesh and compares how well it is aligned to the other mesh. The 'goodness' of a match can be found using some sort of statistical variance to relate how far away each point is from its corresponding point in the other mesh [116]. By obtaining various measures of 'goodness' of match for different offsets, the optimum offset can be found and then used to align the two meshes ready for joining. The drawback with this technique is that it is matching the two meshes by looking at each vertex and not by examining the shape of the lines it is trying to align.

The feature matching technique attempts to align the edges of the two meshes by looking at the shape of the edges and finding common features. Once common features have been found, then one mesh can be scaled and shifted in order that the main features of the face are correctly aligned.

A human operator carries out the configuration of the image capturing system and this means that there are a large number of aspects that can suffer from human error. For this reason, the edges of the meshes produced for each side of the face may not be of exactly the same shape and so a slide matching technique may not be effective. It is therefore decided that a feature matching technique should be used.

In order to use common features of both meshes to join them, it must first be established which features should be used. From the processed images, it can be seen that there are two main features visible, the nose and the mouth. Considering that most of the captured faces will contain these features, they will be used as the reference points for alignment.

It is easy to decide which features should be used for matching, but it is not as easy to write an algorithm to identify these features. In order to make the actual choice of the parameters for identifying the features easier, as much information about the face as possible needs to obtained.

From the edge of the face it is clear that features such as the nose occur at turning points in the curve of the face. A turning point is a point at which the gradient of the curve changes sign. If the curve were a mathematical function, then the turning points could be found using the first derivative of the function, but the curve is made from individual values so a digital technique must be used. This algorithm is defined as follows;

- The points that make up the edge of the face are in an array of n values.
- The gradient for a point can be found by considering the sign (negative, positive or zero) of the equation 5.7.

Approximation to gradient = Value[i] - Value[i+1] Equation 5.33

- When the sign of the gradient has been found for every part of the line, then the sign of the gradient for all adjacent points can be considered. If it is found that the signs are different, then there is a turning point between two values.

The result of these gradient comparisons produces approximate turning points at several places on the line.

Identifying the nose: The nose is represented by the turning point that has the greatest magnitude. It can also be seen in Figure 5.52 that the nose is generally in the centre of the image and that there are turning points below it representing other features of the face. From this definition, the position of the nose can be identified by these simple rules:

1. The nose is the turning point with the largest magnitude.
2. The turning point representing the nose is not the lowest turning point on the line, i.e. nearest the bottom of the face.

Identifying the mouth: the mouth is represented by a series of turning points as shown in Figure 5.52; the top lip, the actual mouth and the bottom lip. It can also be considered that if the nose has already been found, then the turning point that represents it can not be part of mouth. This information produces this set of rules for finding the mouth.

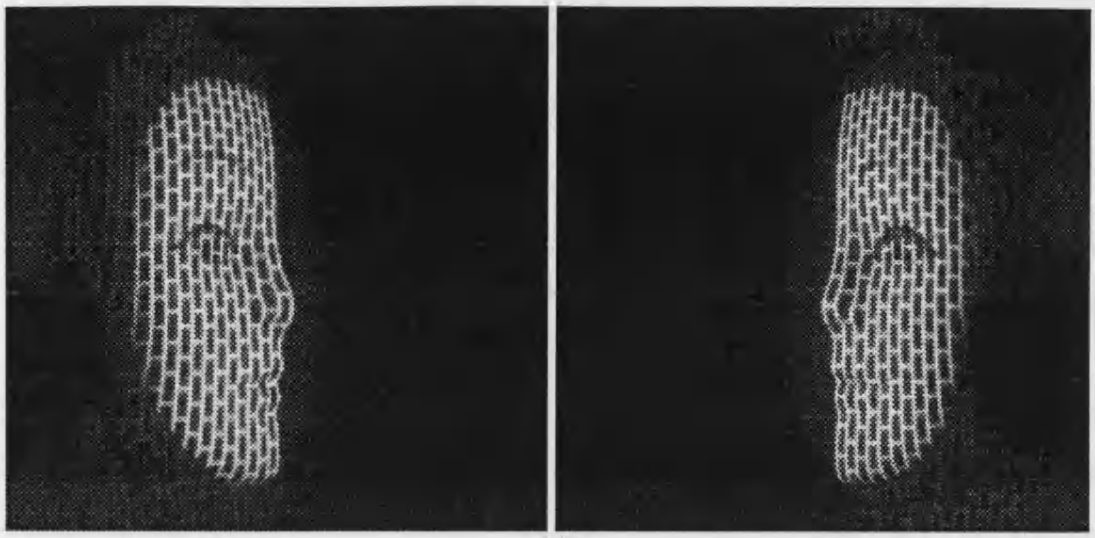
1. The mouth comprises three turning points; a maximum, a minimum and another maximum.
2. The mouth is below the nose and it can not share turning points with the nose.

Two control points on each side of the face to be joined have now been found and the face can be joined using these points as shown in Figure 5.52. The one side of the face (left side) is considered to be correct and the right side of the face is adjusted to line up with it by following the below steps.

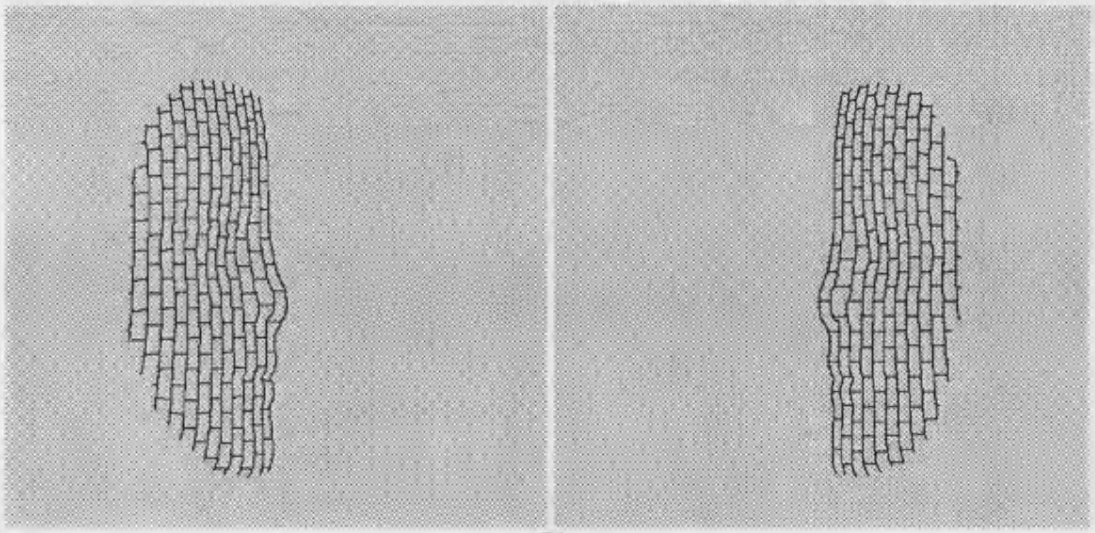
1. Measure the distance between the nose and the mouth for both lines and scale the right line to be the same as the left.

2. Adjust the position of the right line so that its features exactly line up with those of the right.
3. Scale the magnitude of the right line so that the features have the same magnitude.

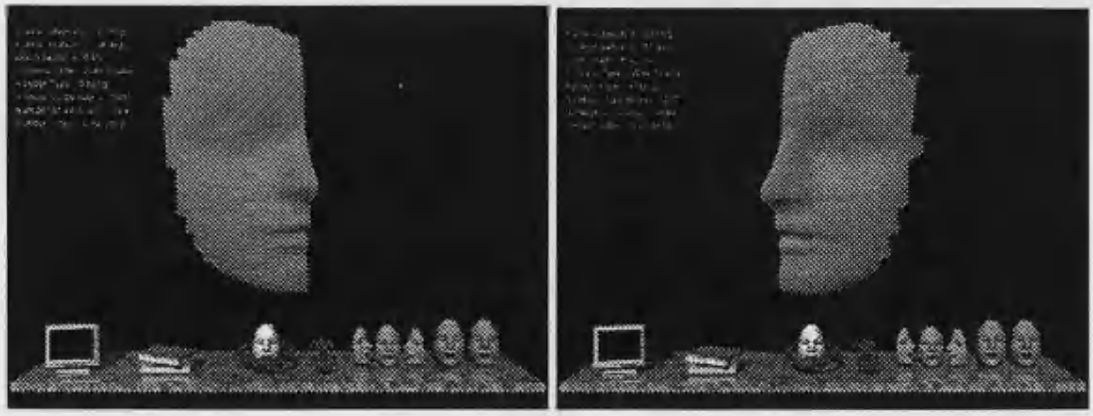
Once the offset and scaling factors have been found, they can be applied directly the vertices of the right side of the mesh to make both sides of the face aligned. The meshes are then joined together to produce a single polygon mesh representing the whole face. The reconstructed image from both sides of a model face is shown in Figure 5.53.



(a)



(b)



(c)

Figure 5.51: (a) Captured images, (b) extracted and corrected images and (c) reconstructed images of the left and right sides of a model face.

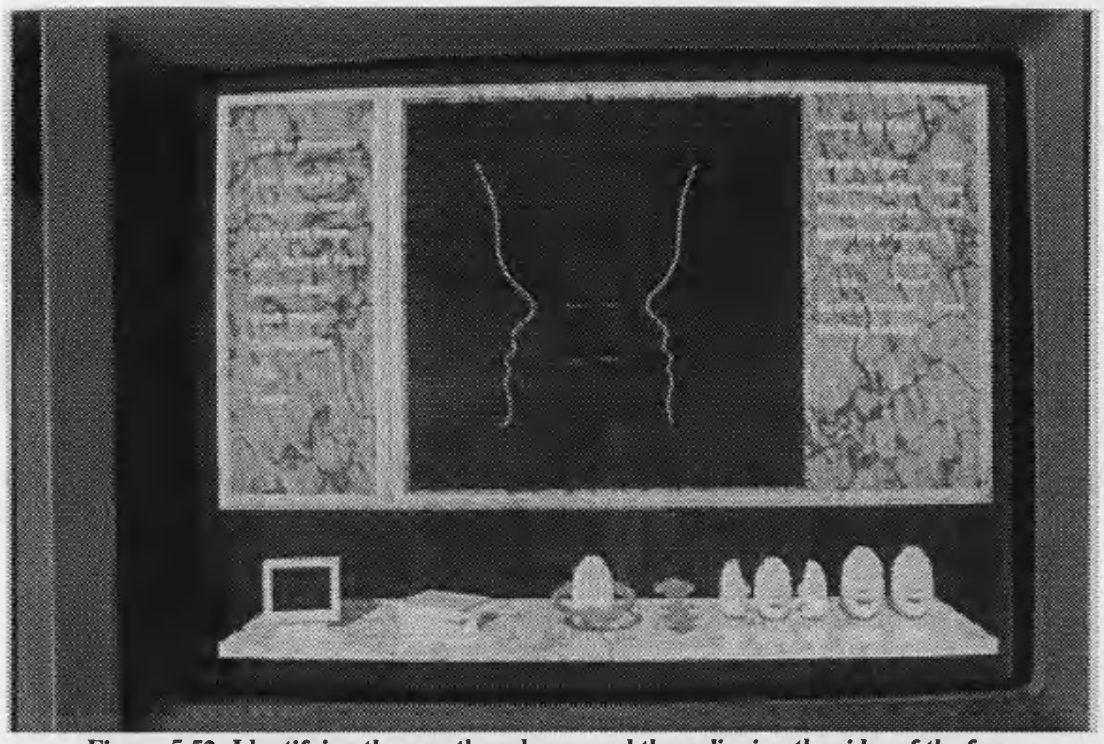


Figure 5.52: Identifying the mouth and nose and then aligning the sides of the face.

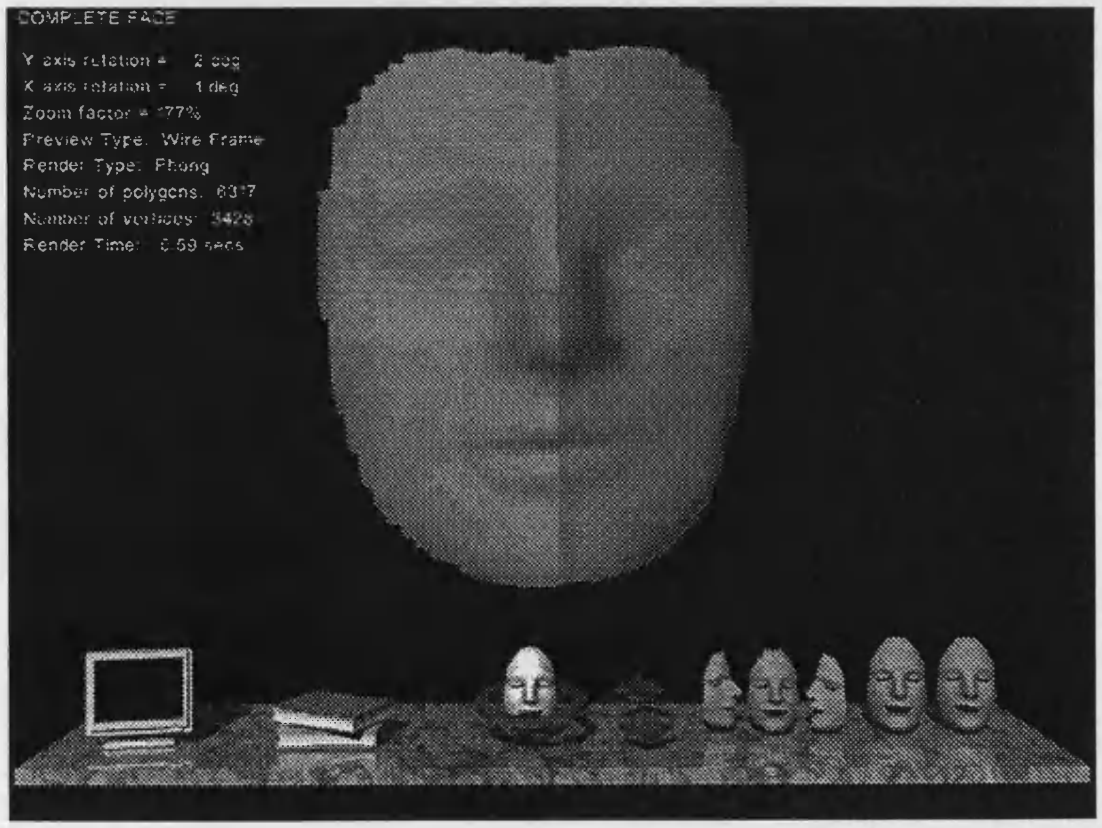


Figure 5.53: A rendered image of a complete face

5.3 Conclusion

A method of matching for a structured light system is proposed by using a marked grid and applying the tracing algorithms. The special characteristic of the implemented algorithms are to correct all of the image problems first and then apply the labelling on a perfect image. This makes the labelling easy and robust in comparison with other implemented techniques based on propagating the defined constraints using the relaxation technique. The selected pattern gives a suitable sequence for the image components so that they can be used for tracing and labelling. Also it was shown that the designed pattern has enough accuracy for capturing and restoring the facial information and therefore extra patterns are not necessary.

The novel tracing algorithms, by applying the inverse difference coding and junction sequences, have been used to extract all of the necessary information from the closed contours on the image. The classification of the problems based on the defined sequences permits the algorithm to predict and correct the deficiencies accurately. The boundaries of the image, i.e. the reference line and boundary network, provide extra information for assessment of the loops during the loop tracing stage. The labelling algorithm is based on line scanning applied on the processed image to index the junctions and vertical lines for the reconstruction stage.

The algorithms for processing the data and ways of presenting that data were reviewed above. The tracing and labelling stages prepared the image for reconstruction, and then a 3-D viewer represented the image in a wireframe or rendered model. The joining algorithm has been developed to join the sides of a face, by identifying the key features. The accuracy of the overall reconstructed image is related to both, the accuracy of the capturing system and patient positioning for capturing the two consistent and accurate images from a face. When the system is complete and accurate, the models for each side of the mesh can be generated, then the algorithm should be able to join the sides of the face accurately.

Chapter 6

Calibration

The aim of calibrating the system is to determine a set of parameters which describe the mapping of projector rays onto the 3-D world co-ordinate space, and the 3-D world co-ordinates onto the image co-ordinates. The process of calibration allows us to calculate the 3-D information of the image and also to accurately determine both camera and projector parameters (position, focal length, orientation, distortions, and so on). Determining the calibration information is complex, and its implementation affects the accuracy of the optic system. Therefore the well known techniques [10, 117 , 118 , 119 , 120 , 121], were reviewed and an advanced model for camera calibration was chosen.

The calibration is carried out in two steps. First, the CCD camera is calibrated in order to establish the transformations between the co-ordinates of the object point in the global co-ordinate system and the co-ordinates of the corresponding image point in the camera recording plane. This is done by observations of a target of known dimensions that is moved through the camera field of view. In the second step the results of this calibration are used to calibrate the projector, by using a calibration target. The projection unit has been suitably modelled and the parameters of the model have been described in the global co-ordinate system.

The camera calibration techniques can be classified in three categories;

1. *Classical Approach*: A large set of non-linear equations which accurately define the world to image plane transform are defined, and large scale optimisation is used to converge on a solution. Although such techniques may include very complex distortions, the results depend on the iterations converging on the correct solution.

2. *Direct Linear Transformation*: Calibration parameters are determined non-iteratively by solving an over-determined set of equations. Until recently all such techniques have used linear techniques so that they were generally fast but inaccurate (since the model cannot correct for non-linear radial distortions). However, *Weng et al.* [120] noted that the direct linear transform (DLT) by *Abdel-Aziz* [22] has been extended to include an inexact correction for radial distortion, and most recently *Shih et al.* [119] have formulated calibration as an eigenvalue problem, providing a solution that is both fast and accurate.

3. *Two-Stage Techniques*: A combination of the above is used to determine a set of parameters using linear techniques before iterating to a final solution. This avoids the convergence problems of (1), and includes the well-known methods by *Tsai* [117] and *Lentz* [118], and a more recent technique by *Weng et al.* [120].

6.1 Camera Calibration

The calibration procedure is to fit the correspondences of a set of known world (3-D) co-ordinates, 'calibration points', and their image co-ordinates onto a camera model. The calibration points consists of markers printed on an accurate three dimension test bed, allowing a 3-D grid of calibration points to be built which will define the world co-ordinate system. The corresponding image co-ordinates of the set of calibration points is determined by (1) automatic image processing to extract the image co-ordinates of the calibration points to one-pixel accuracy, and (2) matching this set of detected co-ordinates with the original set of calibration points. The camera calibration algorithm described below is then used to fit the mapping of world co-ordinates to image co-ordinates onto a camera model. This camera model now allows us to determine the image co-ordinates of any point in the 3-D world co-ordinate system.

At the time of our implementation Tsai's technique was the best available, using a two stage process. The method was modified by using the centre of square calibration targets instead of the Tsai's original model (using the four corners of square calibration targets). Our calibration target, consisting of an 8×8 grid of squares of known spacing to be moved in a direction perpendicular to the plane of the target. Using this arrangement, the image of the calibration target could be taken at different ranges, effectively giving a 3-D grid of points of known world co-ordinates.

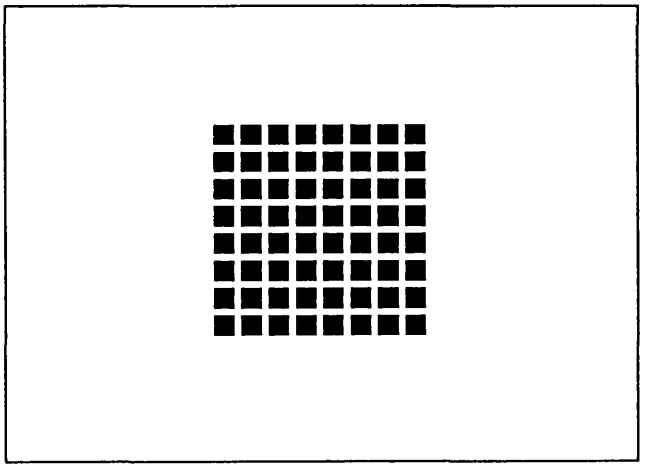


Figure 6.1: Camera calibration target

The camera model used for Tsai's method is shown in Figure 6.2. It defines the calibration parameters and presents the simple radial alignment principle. (x_i, y_i) is the image co-ordinate system. f_c is the distance between image plane and the optical centre. $H_u(x_u, y_u)$ is the image co-ordinate if a perfect pinhole model is used. Due to the lens distortion the actual image co-ordinate is $H_d(x_d, y_d)$. The image co-ordinate in the computer is (x_s, y_s).

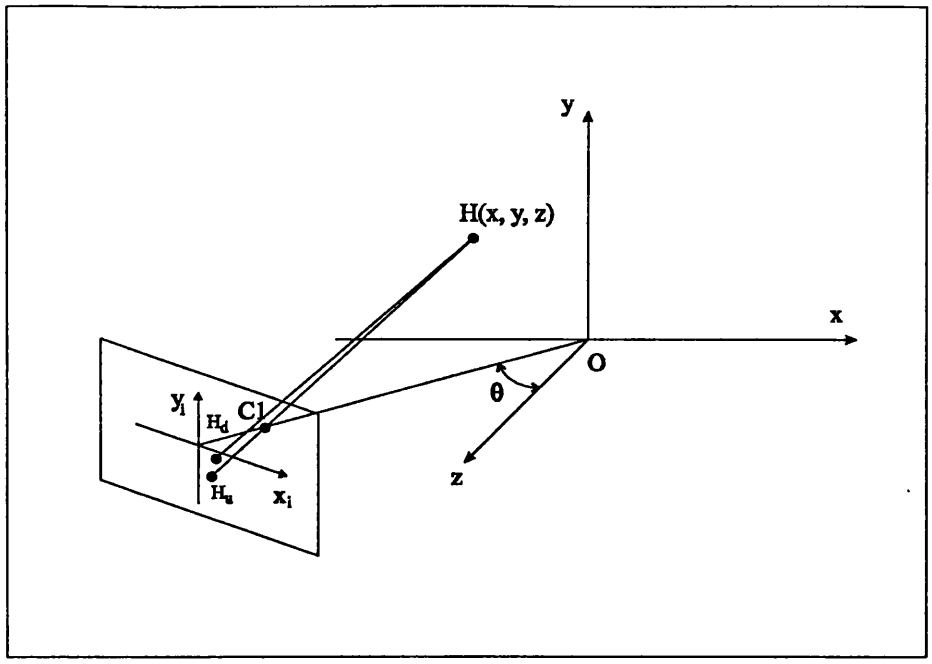


Figure 6.2: The camera model with perspective projection and radial lens distortion

The transformation of point H(x, y, z) to (x_s, y_s) may be done by applying the following steps,

Step 1: Rigid body transformation from the object world co-ordinate system (x, y, z) to the camera 3-D co-ordinate system (x_i, y_i, z_i). The parameters to be calibrated are R (rotation matrix) and T (translation vector).

Step 2: Transformation from 3-D camera co-ordinate (x_i, y_i, z_i) to the ideal image co-ordinate (x_u, y_u). The parameter to be calibrated is the effective focal length f_c.

Step 3: Calculating the radial lens distortion by transferring the ideal image co-ordinate (x_u, y_u) to the distorted image co-ordinate (x_d, y_d). The parameters to be calibrated are distortion coefficients (κ_i).

Step 4: Transformation from real image co-ordinate (x_d, y_d) to the computer image co-ordinate (x_s, y_s). The parameter to be calibrated is the uncertainly image scale factor (s).

The point of intersection of the optical axis with the camera focal plane was found by using the technique described by *Lenz* and *Tsai* [118] whereby the optical axis is determined by shining a laser into the camera lens, knowing that the centre pixel is being sensed when the laser beam is reflected directly back on itself.

6.2 Projector Calibration

The projector calibration is related to the camera calibration results and little prior work has been found on this topic. The main publications by *Keizer et al.* [122] and *Altshuler et al.* [56] describe the linear techniques. The projector is calibrated using the results of the camera calibration. The camera calibration points are removed from the test bed and replaced with a blank sheet onto which the structured light pattern is projected. Since the position of the test bed is known and the camera is calibrated, the world co-ordinates corresponding to each pixel of the detected lines can be determined. This is done by finding the intersection of the corresponding camera rays with the appropriate reference plane match on the test bed. This leads to a set of 3-D feature positions which can fit to a projector model in the camera calibration procedure.

Figure 6.3 shows the projection pattern as a calibration target on a flat screen . As described, the second stage of the calibration procedure is to find the mapping of the projector pattern onto the world co-ordinate space. This is achieved by sampling the position of the projector pattern in world co-ordinates, and fitting this set of samples to a projector model. Since this is the same as camera calibration process we could apply any of the three classes of camera calibration techniques (classical, closed form, or two-stage) to projector calibration. A closed form solution was chosen using a linear model to describe a perspective model for our stripe projector. The model has the correct number of degrees of freedom for the problem. Although our projector calibration algorithm is designed for the parallel line projector system, the same principles could be used for any structured light system by generating a correct model for the projected pattern and devising linear techniques to fit the observed data to this pattern.

Two common types of the projector model were considered, the parallel plane model and the full perspective projector model. A simple parallel plane model is often sufficient to digitise to sub-millimetre accuracy. The full perspective plane model has been used because it provides a fast verification of the orientation of the projector planes and the spacing between the planes.

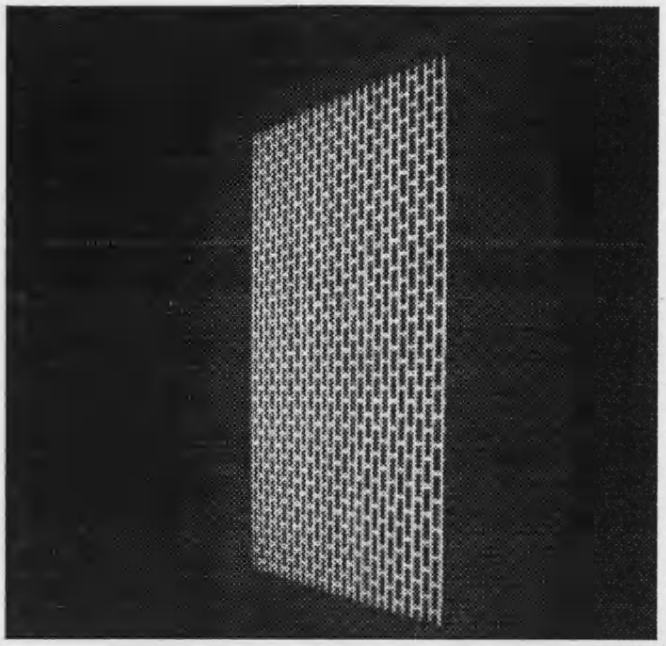


Figure 6.3: Projection of the slide pattern onto a flat screen

6.2.1 Parallel plane projector model

Figure 6.4 shows the parallel plane projector model. Each vertical line is identified by its index e in the original code. The general equation of the light plane e is given by the standard equation for a plane:

$$\underline{n} \cdot \underline{r} = d_e \qquad \text{Equation 6.1}$$

where $\underline{n}$ is the normal to the light stripes, $\underline{r}$ is a point on the plane, and d_e is the distance of the plane e to the origin. The simple linear equation 6.2 is used to define this distance,

$$d_e = d_o.(e - e_o) \qquad \text{Equation 6.2}$$

where d_o is the spacing between adjacent light stripes and e_o is the index of the stripe which passes through the origin (e_o=0). Observed data can be fitted to this model using mean square equation techniques (MSE) [123].

This simple method can cope well where the source data contains some false matches. This allows us to firstly fit all the data to the model, then to filter out the data points that do not closely fit the solution. As long as there was not a catastrophic number errors, a second iteration of the MSE fitting then gives a good fit. The success of the operation can be determined by checking the final MSE of the fit.

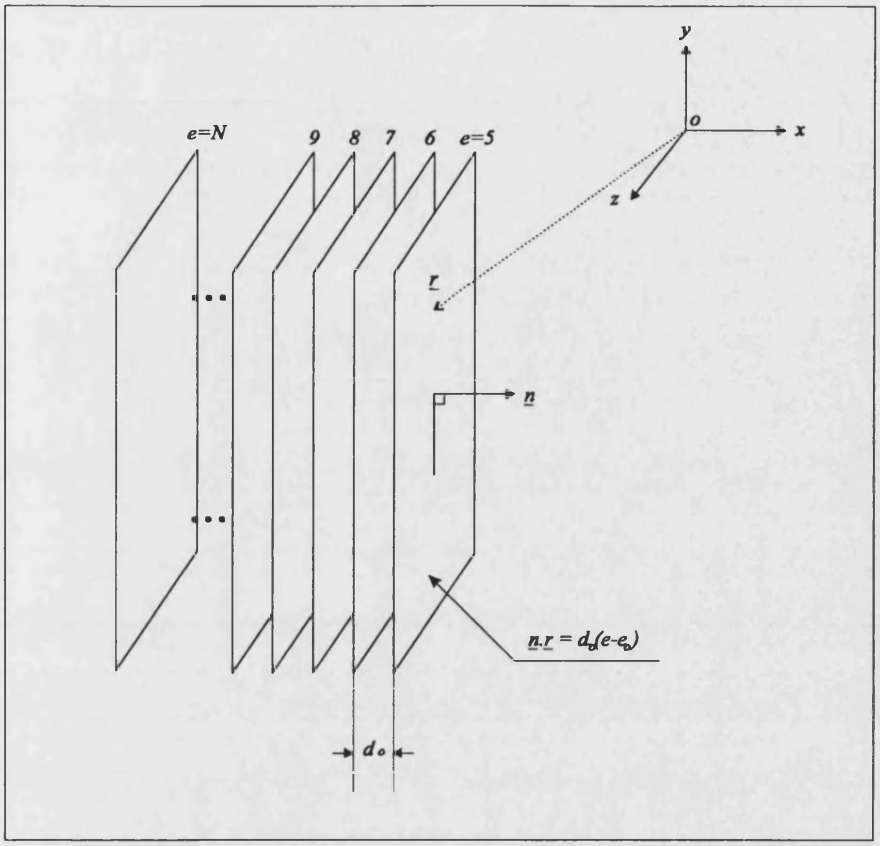


Figure 6.4: The parallel plane projector model. Each plane of light is identified by its index *e*. All planes have normal $\underline{n}$ and are separated by d_o millimetres.

6.2.2 Full perspective projector model

The full perspective projector model is shown in Figure 6.5. Now the normal of each light plane varies, and it is expressed by adding a linear offset to the normal of each plane as a function of the stripe index. This corresponds with a pinhole projection model as explained previously. A normal $\underline{n}_o = \begin{bmatrix} n_x & n_y & n_z \end{bmatrix}^T$ was defined for the line with index 0, and for the light stripe e an offset $e \cdot \underline{p}$ is added where $\underline{p} = \begin{bmatrix} p_x & p_y & p_z \end{bmatrix}^T$, giving the normal $\underline{n}_e = \underline{n}_o + e \cdot \underline{p}$. The light stripes are constrained to pass through the projector focal point $\underline{P}_o = \begin{bmatrix} x_o & y_o & z_o \end{bmatrix}^T$, and the plane equation of the light stripe e is therefore given by the equation 6.5.

$$(\underline{r} - \underline{P}_o) \cdot \underline{n}_e = 0 \qquad \text{Equation 6.3}$$

where $\underline{r} = \begin{bmatrix} x & y & z \end{bmatrix}^T$ is a point on the stripe.

The above parameters are related to the real world dimensions. It can be seen that $\underline{p} \times \underline{n}_o$ defines the axis A along which the projector lies, and that where $\underline{p} \times \underline{n}_e = 0$ (or

equivalent by $e = -(\underline{p} \times \underline{n}_o) / |p|^2$) the light stripe e corresponds to the direction of projector's optical axis.

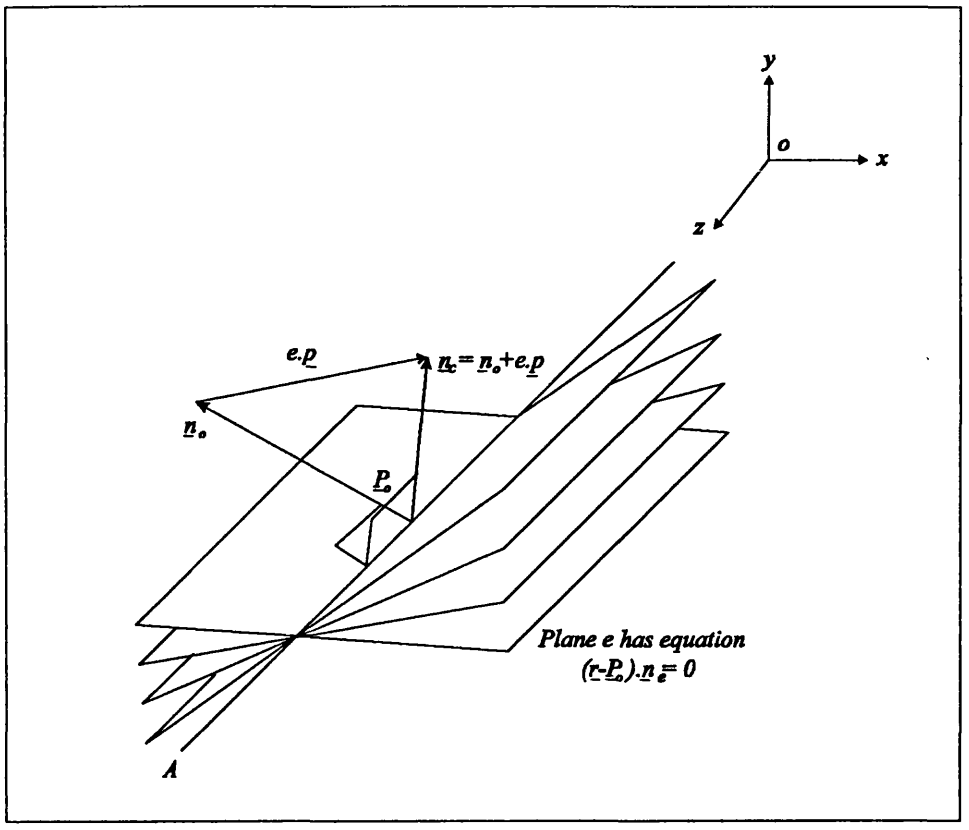


Figure 6.5: The perspective projector model

$\underline{P}_o$ can actually lie anywhere along the axis A. The spare degree of freedom can be removed by choosing $x_o = 0$, which requires that the projector axis is not parallel to the x-axis. This is not a problem in practice since the projector points are along the z-axis so that the z-axis is the normal to the calibration plane. Of course the projector must be allowed to illuminate the plane, and therefore can not be parallel to its x-axis.

$$(\underline{r} - \underline{P}_o) \cdot (\underline{n}_o + e \cdot \underline{p}) = 0 \qquad \text{Equation 6.4}$$

The projector equation 6.4 now is formulated in a form that is suitable for MSE solution. Firstly, writing the equation by using $n_y = 1$, $x_o = 0$:

$$\begin{bmatrix} x \\ y - y_o \\ z - z_o \end{bmatrix} \cdot \begin{bmatrix} n_x + ep_x \\ 1 + ep_y \\ n_z + ep_z \end{bmatrix} = 0 \qquad \text{Equation 6.5}$$

This result can then be rearranged to give the linear MSE problem of the form $A.X = B$, as shown in equation 6.6.

$$\begin{bmatrix} x & xe & y & ye & z & ze & e & 1 \\ x & xe & y & ye & z & ze & e & 1 \\ & & & \cdots & & & & \\ & & & \cdots & & & & \end{bmatrix} \cdot \begin{bmatrix} n_x \\ p_x \\ e_o \\ p_y \\ n_z \\ p_z \\ -(y_o p_y - z_o p_z) \\ -(y_o - z_o n_z) \end{bmatrix} = \begin{bmatrix} -y \\ -y \\ \cdots \\ \cdots \end{bmatrix}$$

Equation 6.6

The term -y was moved to the right side of the equation in order to avoid the solution of $\underline{n} = \underline{p} = \underline{P}_o = e_o = 0$. The solution matrix X gives e_o, $\underline{n}$ and $\underline{p}$ directly and the two elements in the last row can be solved simultaneously to give y_o and z_o. The equation of light stripe e can then be expressed as $(\underline{r} - \underline{P}_o) \cdot \underline{n}_e = 0$, or using the usual form for the equation for a plane $\underline{r} \cdot \underline{n}_e = \underline{P}_o \cdot \underline{n}_e$. The normal $\underline{n}_e$ can be normalised so that the right hand side of the plane equation is the closest distance from the origin to the plane. The result of the calibration can be detected by checking both the MSE of the fit and the values for $\underline{P}_o$ and the projector axis.

6.3 Results and Conclusions

The calibration procedures were used for the facial image 'Face', Figure 5.20. The camera calibration set-up used a calibration target consisting of 8×8 squares laser printed at 300dpi, at 10 different ranges, giving 640 calibration points respectively. The dimension of the printed squares array were measured over 7 intervals to an accuracy of 0.1/7 mm and the range was determined to an accuracy 0.01 mm, using a vernier for all measurements. The squares were detected in the camera images by using auto extraction algorithm and their centres of gravity were used as the calibration points.

Table 6.1 shows the result of the camera calibration for 'Face'. The pixel errors of the fit to the camera calibration model are translated into errors in millimetres.

Distance between the squares (centre)	6.5 mm
Number of calibration planes and ranges	15: (10,...,150 mm)
Maximum absolute error of fit to model	1 mm

Table 6.1: Camera calibration input and result

The calibration does not currently consider the different positions of different fields on the camera focal plane. The results for the images corresponding to one field only has been shown.

The projector calibration technique with a closed-form solution was chosen, using a linear model to describe a one dimensional perspective model. It was found that by using this type of projector model (with a long projector focal length so that radial distortion is minimised), the errors involved in fitting the calibration data to the model are greater than in the case of camera calibration. The increased error primarily is related to the fact that the projector calibration depends on the result of the camera calibration, and is therefore subject to its inaccuracies. A second reason for the difference is that whereas it can be expected that the CCD camera is highly engineered and the dimensions of the CCD array are well defined and regular, a commercially slide projector may contain non-linearity, and may produce distortions into the projected pattern.

In order to measure the accuracy of the projector calibration, structured light images of planes at known positions in world co-ordinates were digitised and processed by using the result of the above calibration. The distribution of the errors between the measured positions and the true positions of the planes in world co-ordinates was determined, and is shown in Table 6.2. The estimated ranges to the projector correspond with the manual measurements within the accuracy of it, indicating that the model corresponds well with reality.

Standard deviation	0.5 mm
Projector to origin distance	700-750 mm
Manual estimate (±1mm)	700 mm

Table 6.2: Perspective model projector calibration results

Using the results of the calibration it is possible to determine the system resolution (which is defined by the geometry of the optical set-up, the resolution of the camera and the line sampling interval), and to estimate the focal length of the lens.

The resolution of the 3-D data samples is considerably better than 1 mm in most cases. Additionally, the calculations assume single pixel feature accuracy, and as shown in chapter 5 our feature extraction algorithm achieves considerably very good pixel accuracy.

In order to show that the results of the calibration corresponded well with reality, checks were carried out on the camera/projector separation angle, the focal length, camera to subject range and projector to subject range. The results are shown in Table 6.3. The Equation defined in Appendix B was used to estimate of the focal length, assuming that the camera is focused in the centre of the working volume. The result corresponded well with the true focal length (4 mm). Finally the camera to subject range was accurate within the accuracy of the manual measurement.

f_c	4.03 mm
θ	36°
Line sampling interval	2 mm

Table 6.3: Derived calibration information

Since the structured light system can cope with arbitrary camera and projector orientations, strict alignment of the video equipment is not required. It is only recommend that the camera and projector optical axis be approximately co-planar, and that the vertical axes of the camera and projector focal planes be approximately parallel. This means that shadowing between the camera and projector in the vertical plane is minimised.

Chapter 7

Performance and Results

7.1 Performance

The performance of the system [124] as a 3-D computer vision system is summarised here. The strongest influence on these performance measures are indicated by the lines in Figure 7.1, leading from boxes describing the implementation of the system. Briefly, the implementation of the algorithms affects the 3-D accuracy of the system, the optic system affects the accuracy, as well as the amount of shadowing between the camera and projector (occlusion angle) and the resolution. The system hardware affects the resolution (from the CCD array resolution), stand-off (from the optics), repeatability (from the noise sources such as line jitters), and measurement rate.

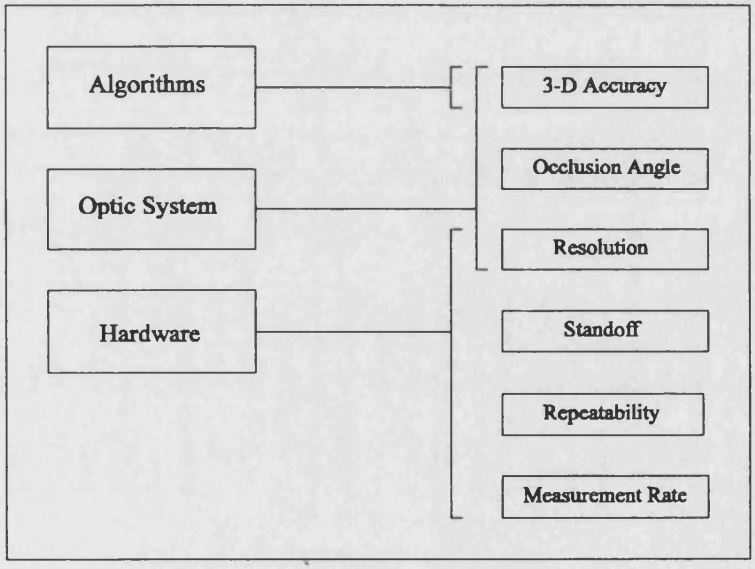


Figure 7.1: Factors affecting system performance

These performance measures are covered below:

1. Accuracy: The accuracy of the feature extraction and interpretation is sufficient for recovering the sampling data. Also the 3-D map for generating the 3-D viewer has been designed to sample accurately all the pixels of the lines for reconstruction.
2. Occlusion angle: The occlusion angle is simply the angle between the camera and projector.
3. Resolution: The surface resolution of the image is about 1 mm for a whole face, and this can be improved by applying the system to a localised areas of the face such as mouth, nose and etc.
4. Stand-off: The stand-off is determined by the optics of the system. For the camera 4 mm lens the closest focusing distance is approximately 300 mm.

5. Repeatability: The repeatability depends on factors such as line jitters and equipment noise level. A large number of images have been captured and processed by the system during the last two years. The quality of images are good so that noise is negligible.
6. Measurement rate: The processing time for the hardware and algorithms are less than 5 minutes. It should be noted that most of the algorithms are not optimised for speed.

The system, as do many vision systems, suffers from limited depth of field, which could be overcome by using a wider camera lens and a brighter projector. A projector brighter than the 250 W model is necessary. As it was, the range to the subject, the camera/projector separation angle and the optics were adjusted until the structured light images were sharp. The subject was required to remain immobile within the image capturing stage.

Poor contrast level from different skin colour and any hair on the face can cause problems during the image processing step. It can be argued that for maxilla-facial surgery, like other major operations on face, the face can be shaved and prepared. Alternatively, the application of a harmless lotion or powder to the face could compensate for the poor contrast, the advantage of this method being that it is a non-invasive technique.

To evaluate the performance of the system, the following tests have been carried out.

1. The effect of discontinuities: sharp nose and hole.
2. Performance for realistically shaped surfaces: using real faces or subjects.
3. Confusing scenes: using surfaces with concave and convex areas.
4. Equipment related effects: Depth of field and added noise.
5. Surface colour and intensity variations. Different type faces such as Caucasian, Asian and European.

The algorithms for processing the data and ways of presenting that data were reviewed in chapter 5. The tracing and labelling stages prepared the image for reconstruction, and then a 3-D viewer represented the image in a wireframe or rendered model. The joining algorithm has been developed to join the sides of a face, by identifying the key features. The accuracy of the overall reconstructed image is related to both, the accuracy of the capturing system and patient positioning for capturing the two consistent and accurate images from a face. When the system is complete and accurate, the models for each side of the mesh can be generated, then the algorithm should be able to join the sides of the face accurately. Now the reconstructed images will be assessed by considering three main elements of the overall accuracy; calculated height, reconstructed image and deformed faces.

The height calculation algorithm was applied to the test objects with a known shape to check the accuracy. A flat plane was used to calibrate the system and then the accuracy tested with a 45° ramp to the base plane and a spherical object with known surface equation as shown in Figure 7.2. The optic system parameters were θ=45°, d_p=725 mm, d_c=285 mm.

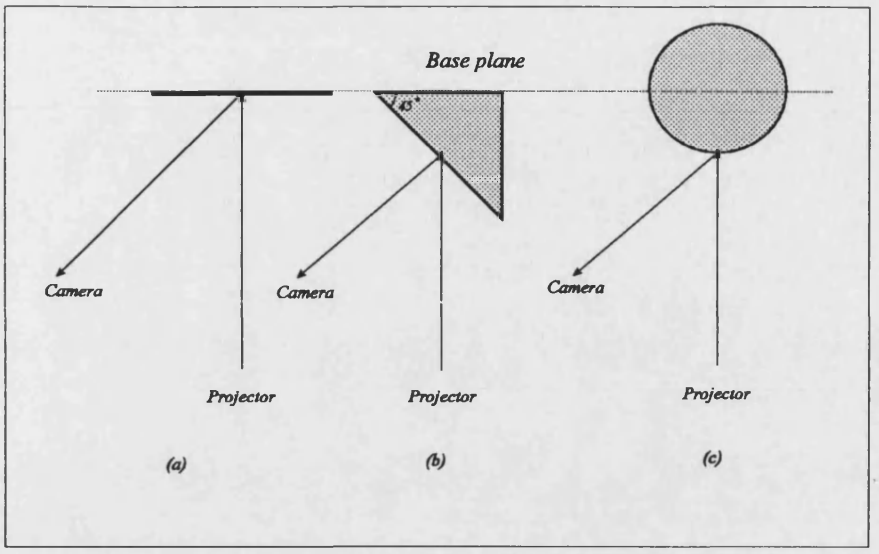
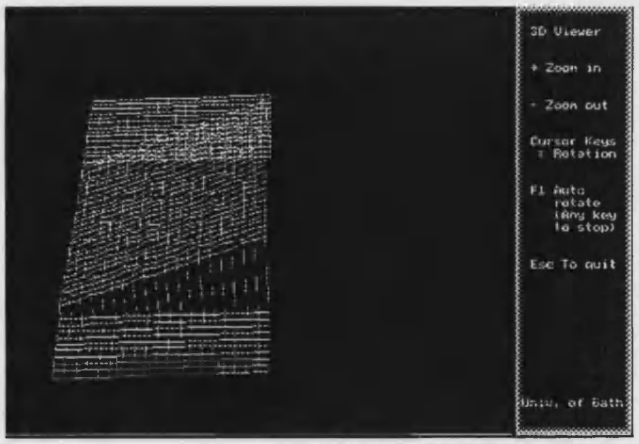
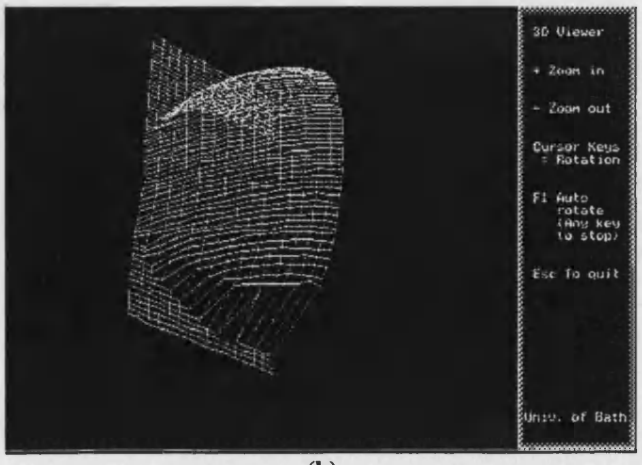


Figure 7.2: (a) A flat plane for calibrating the height and 3-D mesh , (b) a ramp and (c) a spherical object for testing the height accuracy.

The reconstructed images of the test objects are shown in Figure 7.3.



(a)



(b)

Figure 7.3: Reconstruction of the test objects for evaluating the height accuracy, (a) a ramp and (b) spherical object (a ball)

The results were compared with the measured values and the absolute errors, |*Calculated_value* - *Measured_value*|, were found as shown for only one row (y=250) in Table 7.1.

Line	*x`*	*y`*	*Calculated height (mm)*	*Measured height (mm)*	*Absolute error (mm)*
1	111	250	1.8	1	0.8
2	119	250	9.48	9	0.48
3	127	250	17.5	17	0.5
4	135	250	25.4	25	0.4
5	142	250	32.6	32	0.6
6	150	250	40.65	40	0.65
7	157	250	47.6	47	0.6
8	165	250	55.7	55	0.7
9	173	250	63.5	63	0.5
10	180	250	70.45	70	0.45
11	188	250	78.7	78	0.7
12	195	250	85.6	85	0.6
13	203	250	93.55	93	0.55
14	211	250	101.75	101	0.75
15	218	250	108.8	108	0.8
16	226	250	116.75	116	0.75
17	234	250	124.85	124	0.85
18	241	250	131.85	131	0.85
19	249	250	139.9	139	0.9
20	257	250	147.95	147	0.95

(a)

Line	*x`*	*y`*	*Calculated height (mm)*	*Measured height (mm)*	*Absolute error (mm)*
1	129	250	3.39	3	0.39
2	141	250	22.55	22	0.55
3	154	250	42.23	42	0.23
4	166	250	57.2	57	0.2
5	179	250	70.63	70	0.63
6	189	250	80.21	80	0.21
7	201	250	90.66	90	0.66
8	212	250	99.83	99	0.83
9	223	250	105.53	105	0.53
10	234	250	112.52	112	0.52
11	245	250	118.97	118	0.97
12	255	250	124.81	124	0.81
13	265	250	128.73	128	0.73
14	274	250	133.62	133	0.62
15	283	250	137.82	137	0.82
16	292	250	140.95	140	0.95

(b)

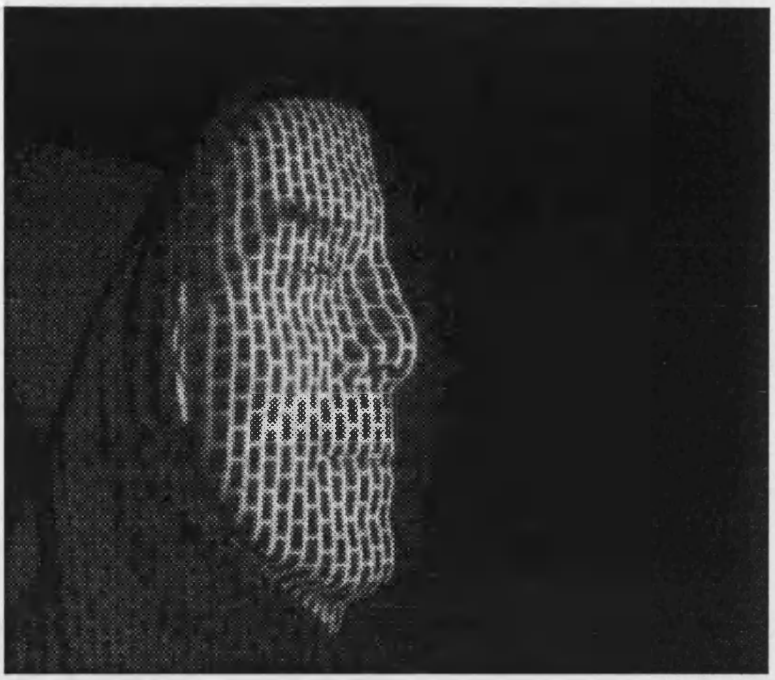
Table 7.1: Height calculation for (a) a ramp, (b) a ball

The absolute error is less than one millimetre and it shows that the height equation has sufficient accuracy for evaluating the reconstructed facial images.

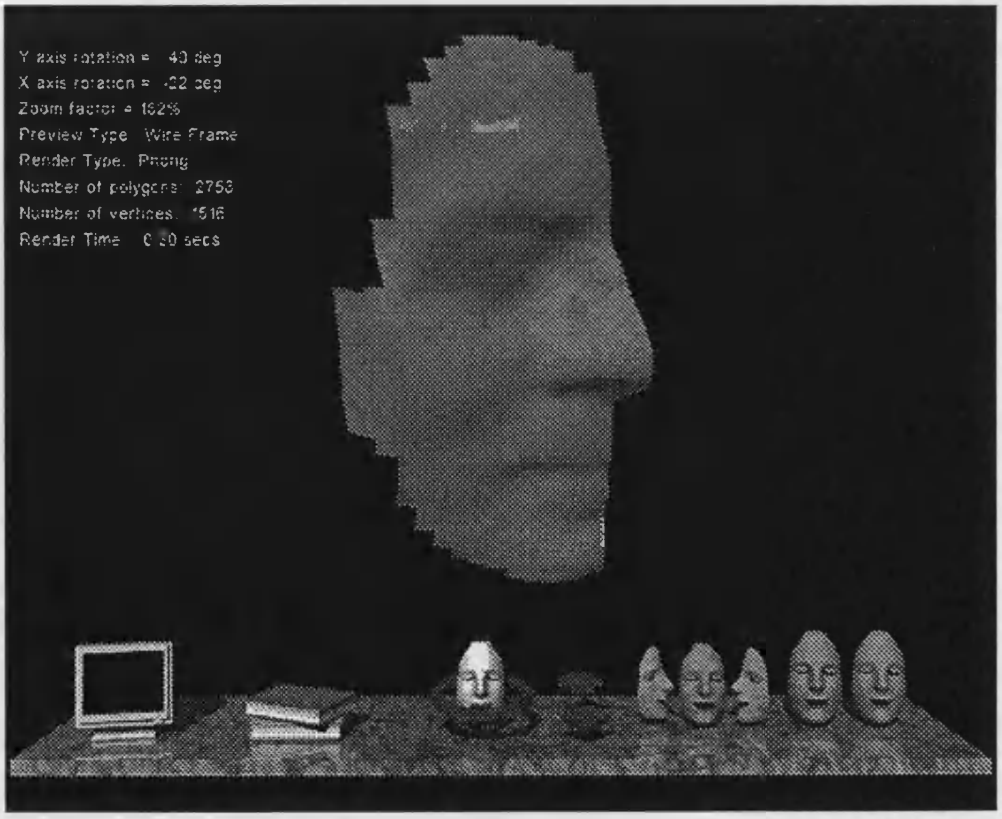
The next section presents the results of the processed images on many different candidate faces.

7.2 Results

The display method for a reconstructed surface was improved by using shading and high resolution graphics. Also the program environment of the software has been converted to a fully mouse controlled system which makes it easy to use by non-experienced users. To show that the implemented system and algorithms have enough accuracy for processing real faces, different kinds of images from the candidate faces were taken and processed. The extraction, interpretation and reconstruction of the images show that the written algorithms have appropriate accuracy on real faces. The captured and reconstructed facial images from the candidate faces are shown in Figure 7.4 -Figure 7.10.

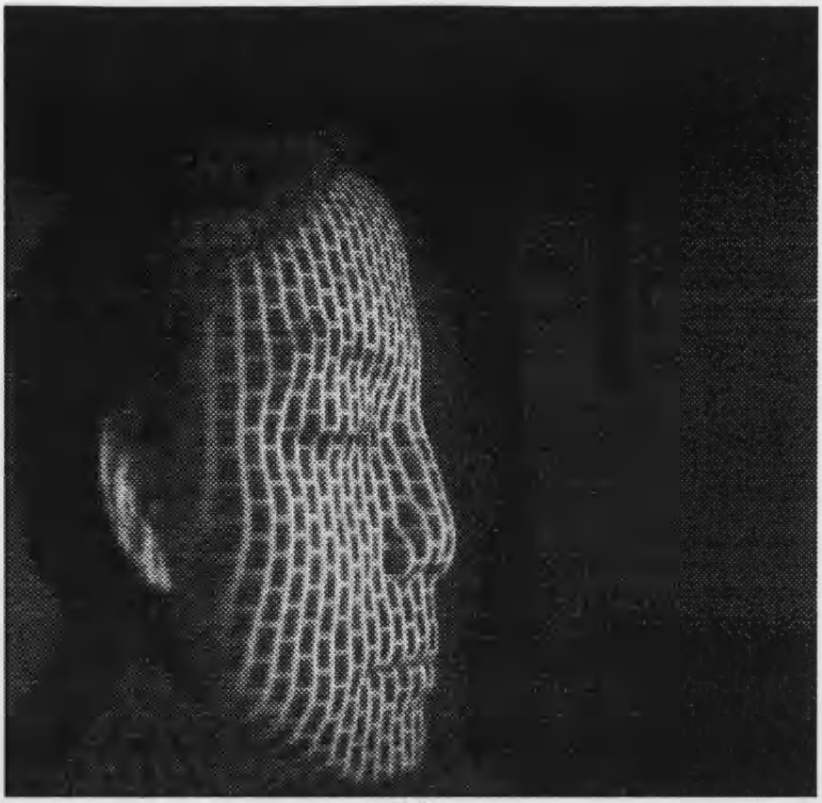


(a)

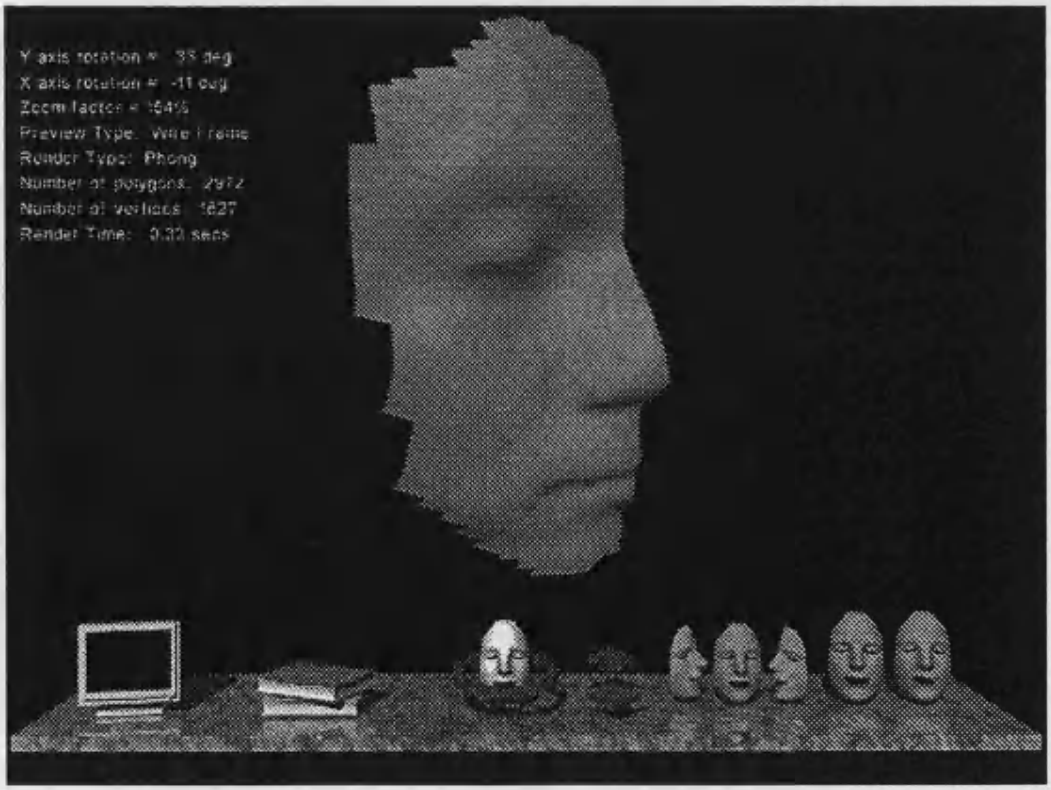


(b)

Figure 7.4: (a) Captured image and (b) reconstructed image of the European man 'Dave'

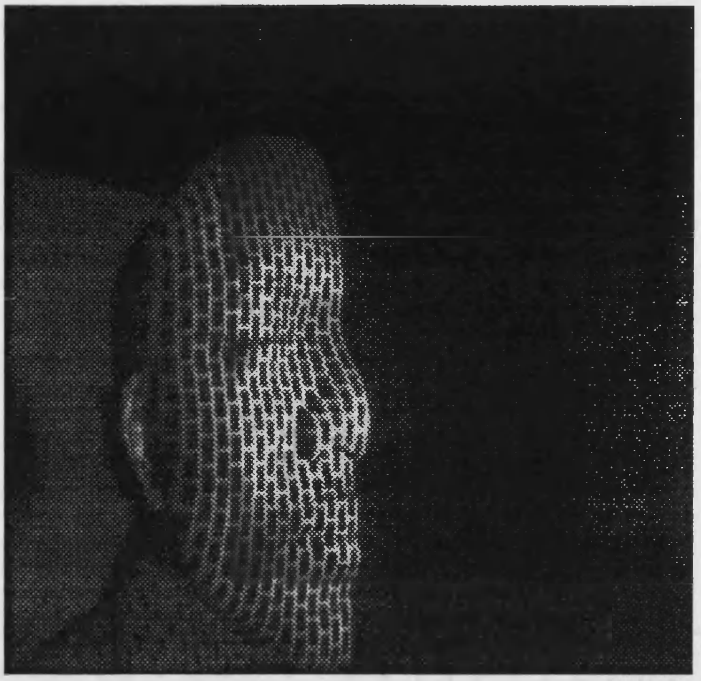


(a)

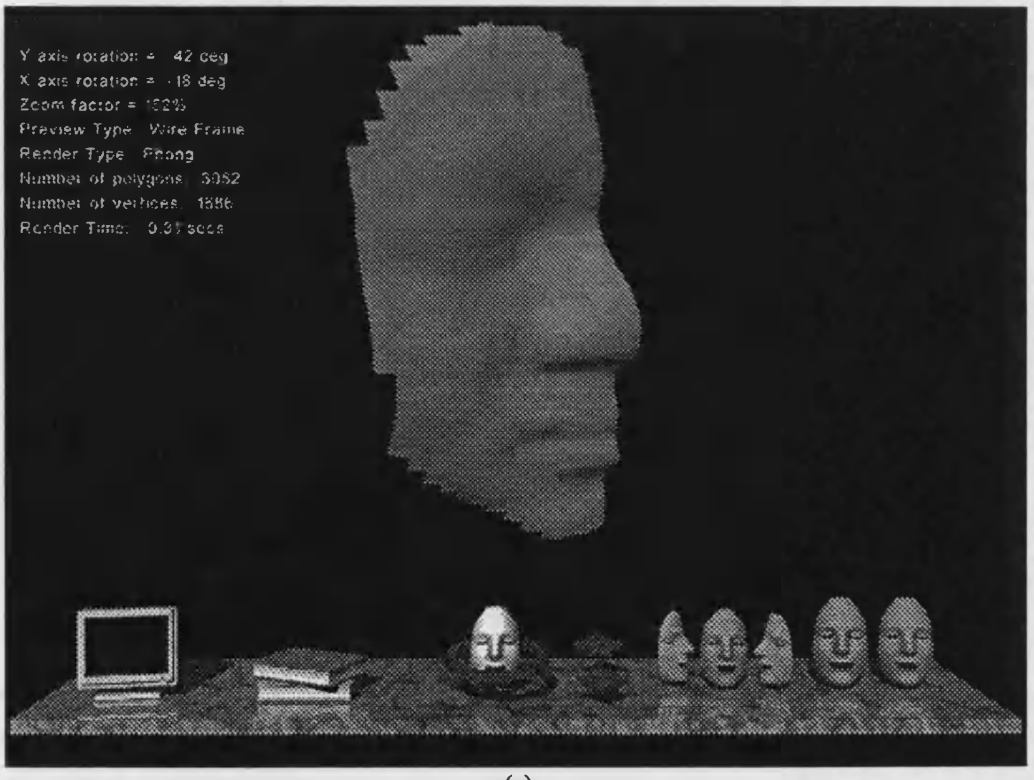


(b)

Figure 7.5: (a) Captured image and (b) reconstructed image of a Chinese man 'Jin-Pen'

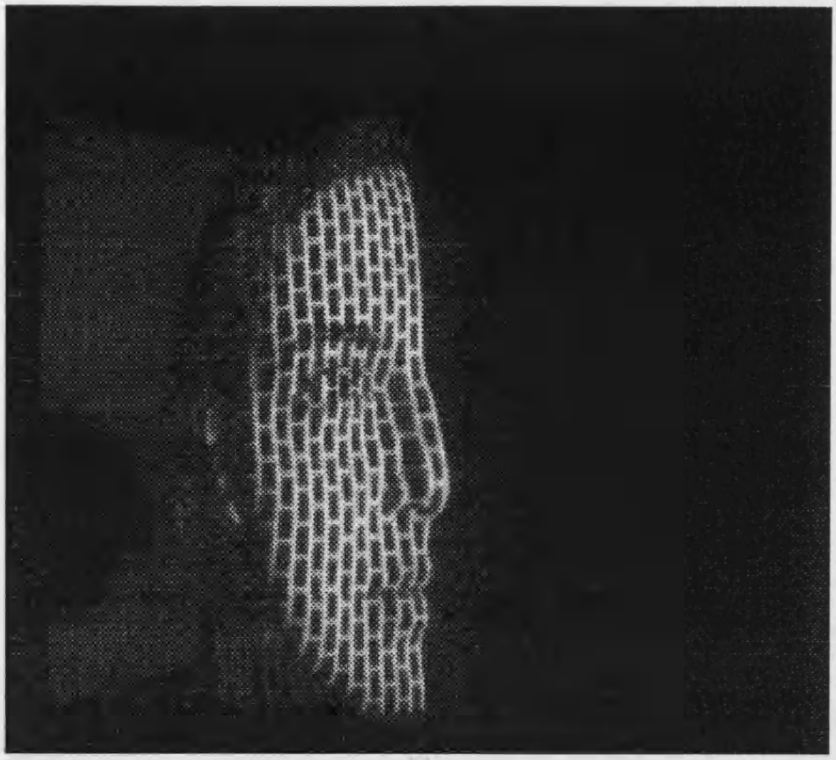


(a)

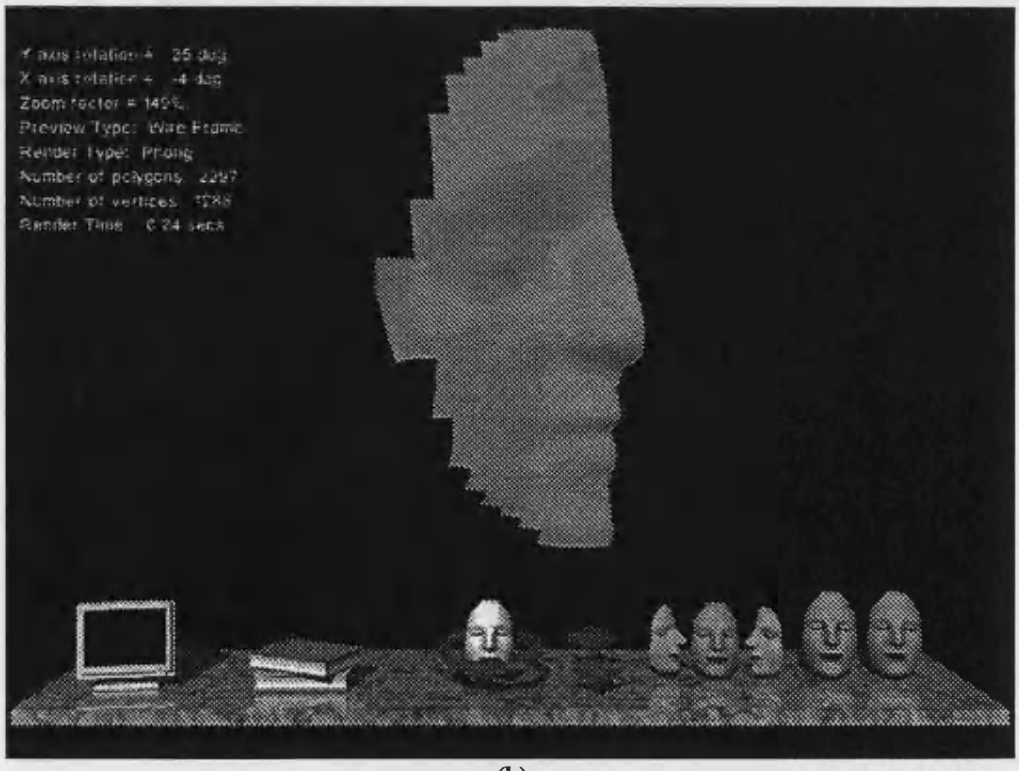


(c)

Figure 7.6: (a) Captured image from the African face 'Ola', and (b) reconstructed image.



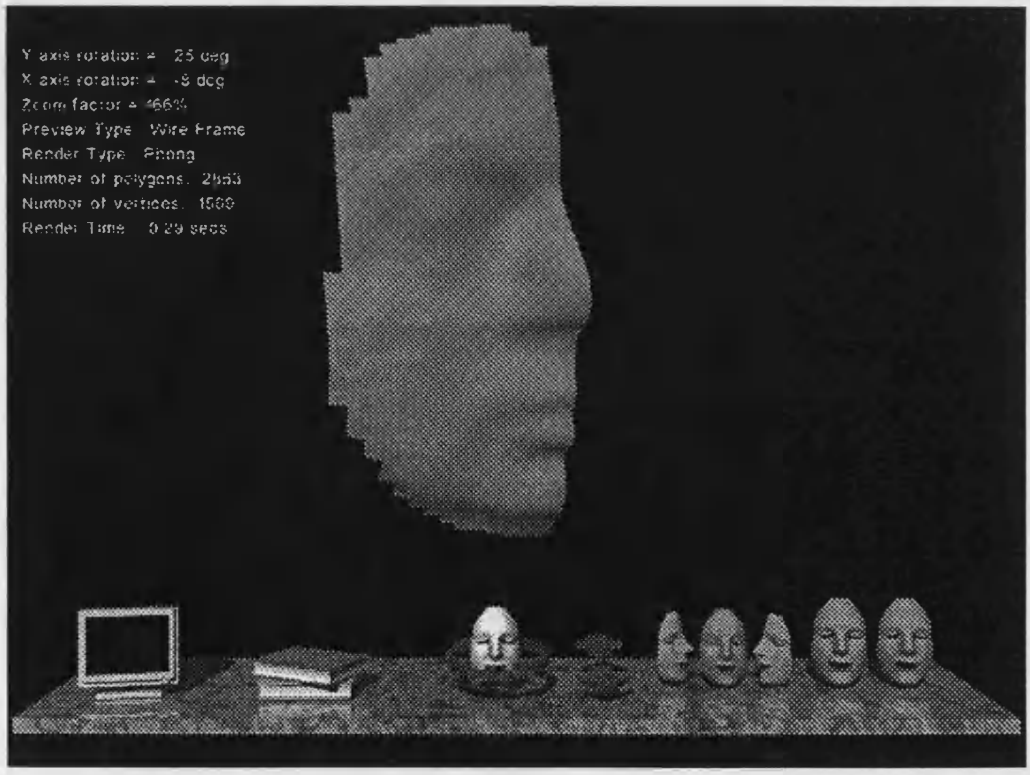
(a)



(b)

Figure 7.7: (a) Captured image and (b) reconstructed image from a lady 'Federica'

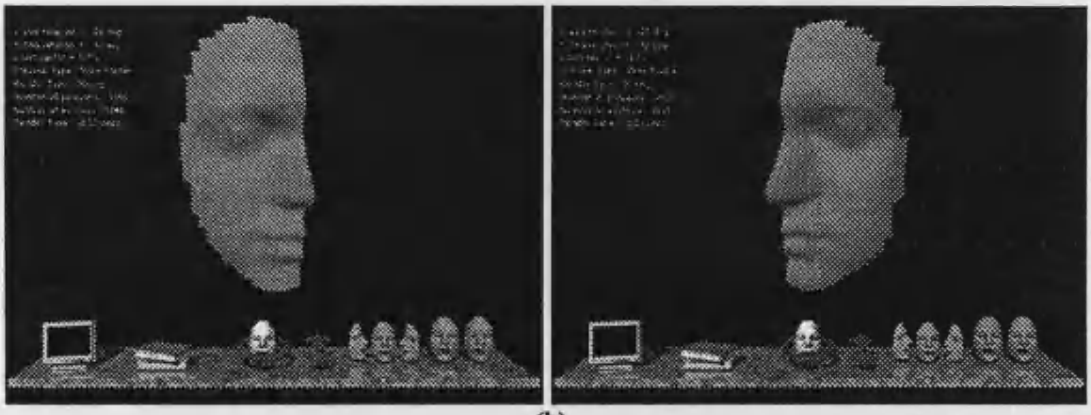
(a)



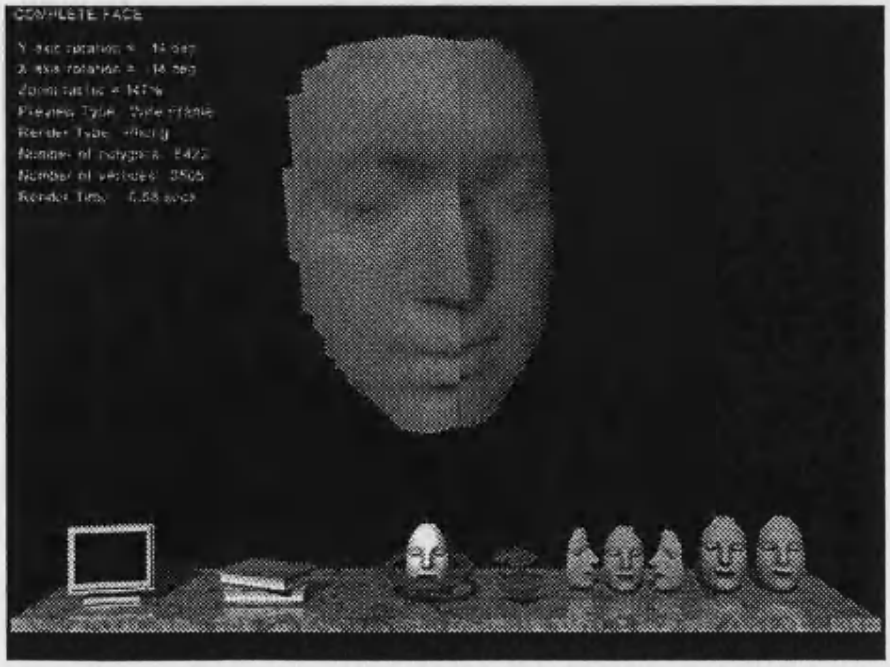
(b)

Figure 7.8: (a) Captured image and (b) reconstructed image from a Turkish man 'Serdar'

(a)

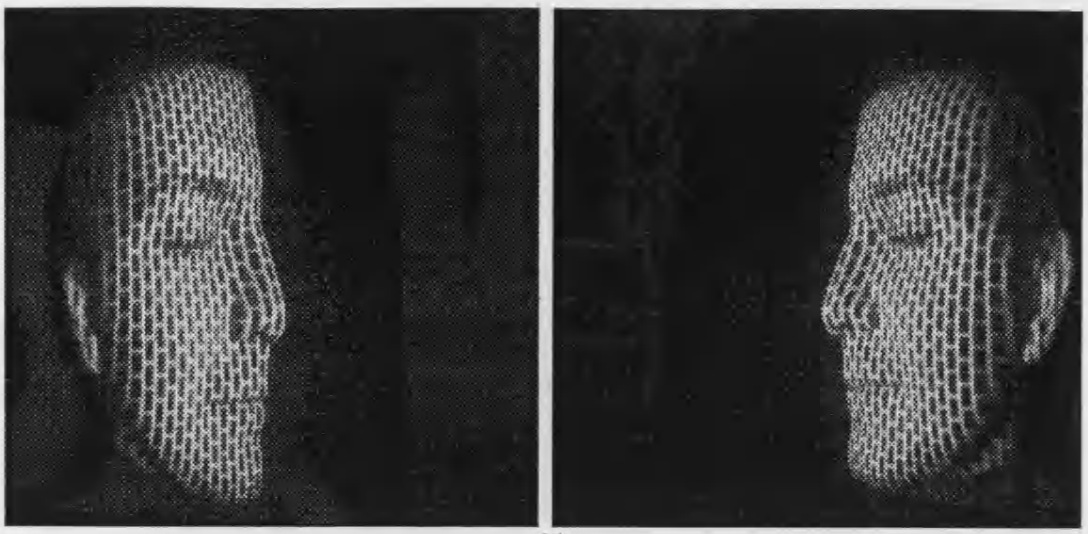


(b)

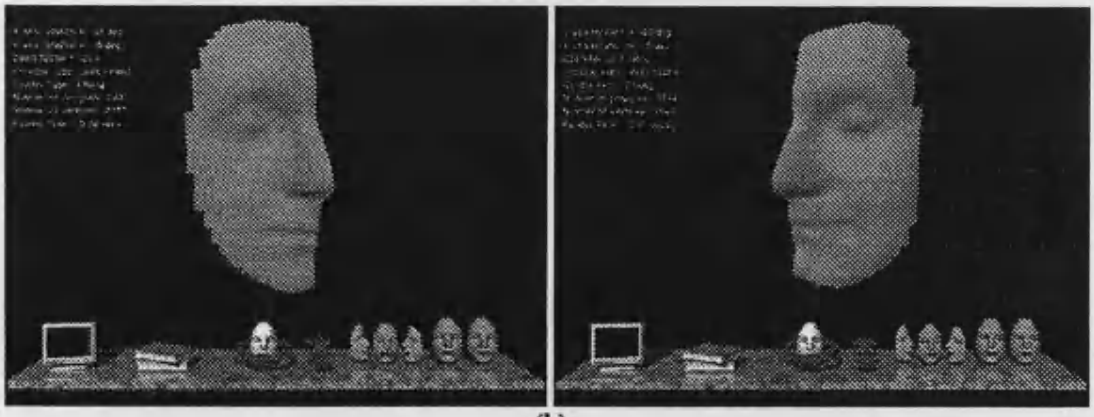


(c)

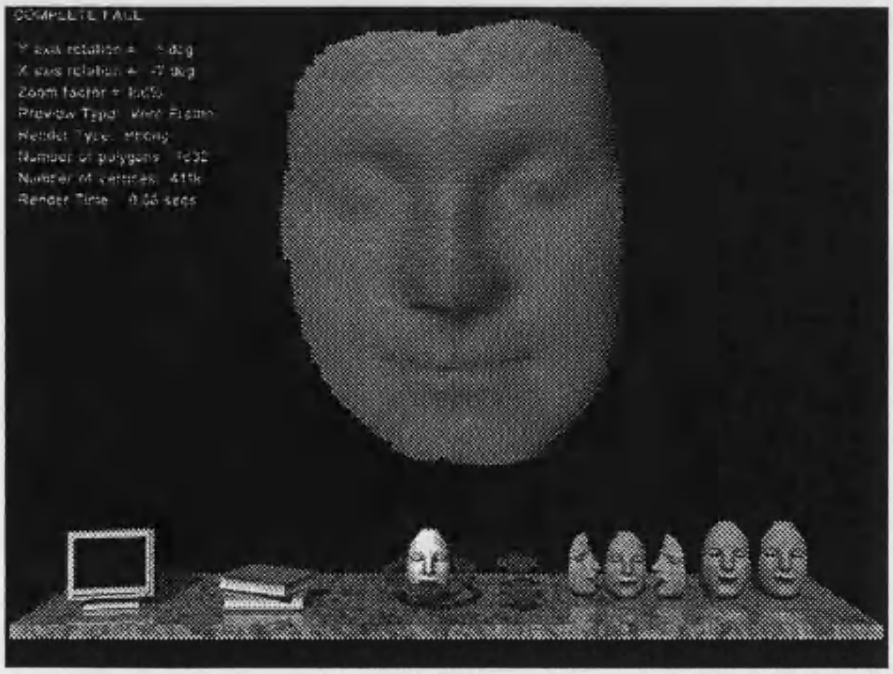
Figure 7.9: (a) Captured images from the left and right side of the face 'Nick', (b) reconstructed images and (c) a complete face.



(a)



(b)



(c)

Figure 7.10: (a) Captured images from the left and right side of the face 'Nicola', (b) reconstructed images and (c) a complete face.

The analysis of the facial surface is done by decomposition of the surface into regions of fundamental surface types which are related directly to the terms in common use for describing the face such as nose, mouth or eyes. Some important and measurable points on the face are used to show the 3-D accuracy as shown in Figure 7.11.

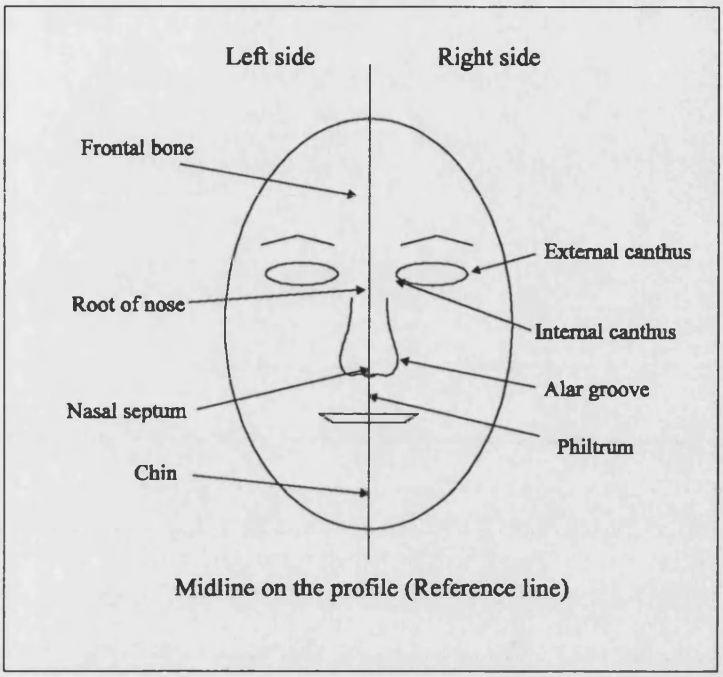


Figure 7.11: How the face is divided to measure the surface regions.

The results of calculating and direct measuring of the height for the high lighted points on the reconstructed 'face', Figure 5. , from the base plane (on the patient positioning) are demonstrated in Table 7.2. The results from direct measurement and calculation show that the calibration and reconstruction stages produce enough accuracy.

The actual distance between two points on a flat surface is calculated from Pythagoras's theorem as follows,

$$Actual\ three\text{-}dimensional\ distance = \sqrt{distance^2 + depth^2} \qquad \text{Equation 7.1}$$

where distance is the two dimensional measurement from the structured light image and depth is the line depth between points. If the points are selected from the two adjacent lines on a face, then equation 5.11 can be extended to calculate the selected distance on the surface. The 3-D map includes the height information for the structured light points on the subject's face so it is suitable to use for calculating the distance of a selected points and then calculating the areas.

Measured area	Calculated value from the structured light and reconstructed image (mm)	Direct measurement from the face (mm)	Error (mm)
Height from the frontal bone to the base plane	249.1	250	0.9
Height from the root of nose to the base plane	244.5	246	1.5
Height from the left internal canthus to the base plane	223	225	2
Height from the left external canthus to the base plane	189.4	191	1.6
Distance between the left internal canthus and external canthus	52	54	2
Height from the nasal septum to the base plane	268.6	271	1.4
Height from the alar groove to the base plane	236	238	2
Height from the philtrum to the base plane	241.8	243	2.2
Height from the chin to the base plane	254.2	256	1.8

Table 7.2: Accuracy of the structured light system on the reconstructed 'face'

Errors in measurement of distance should be small using vernier callipers. A great source of error is landmark identification for a real face. Failure to reproduce the exact location means that not only the two-dimensional measurements between points changes but the height estimate can be affected. Also extremes of facial expression such as pouting or laughing can obviously affect the position of landmarks such as the mouth corners located on mobile soft tissues.

Now to show that the system can be used for different type of faces, a series of the test images with predefined deformities were selected. These deformities were applied onto the dummy heads by considering both concave and convex problems on a face, and the reconstructed results were compared with the original deformed faces.

Figure 7.12 shows the left side of a model face with a lump on the cheek near the mouth. The height for any point on the surface can be determined directly from the reconstructed image but the distance and area may be calculated indirectly by considering the 2-D structured light image [28].

The calculated and measured values for the lump are shown in Table 7.3.

Measured dimension	Calculated value from the structured light image	Direct measurement from the model head
The maximum height of the lump from the facial surface	13.3 mm	15 mm
Surface area for the lump	1445 mm^2	1474 mm^2

Table 7.3: The measured and calculated values for the height and area of the lump.

The implemented system also has sufficient accuracy to analysis the concave areas on a face. Figure 7.13 demonstrates a dummy head with two concave areas on the forehead and cheek.

Both cavities are evaluated and the relative information calculated as shown in Table 7.4.

Measured dimension for the cavity on the forehead	Calculated value from the structured light image	Direct measurement from the model head
Depth	13.7 mm	15 mm
Area	1224 mm^2	1240 mm^2

(a)

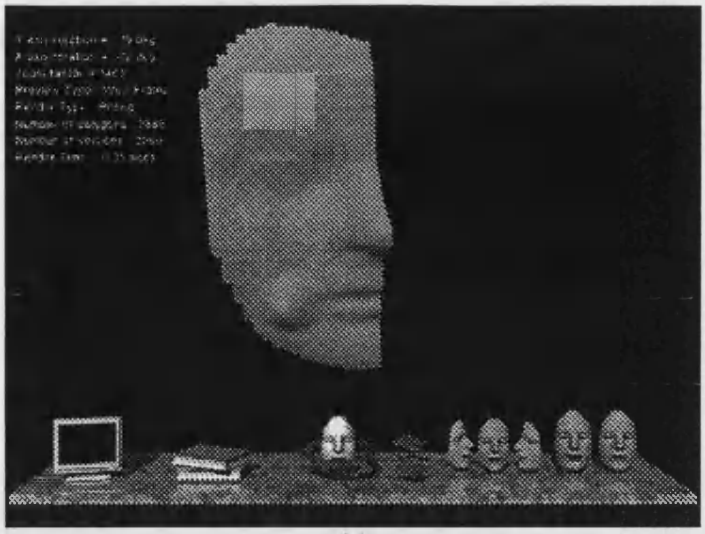
Measured dimension for the cavity on the cheek	Calculated value from the structured light image	Direct measurement from the model head
Depth	8.6 mm	10 mm
Area	401 mm^2	412 mm^2

(b)

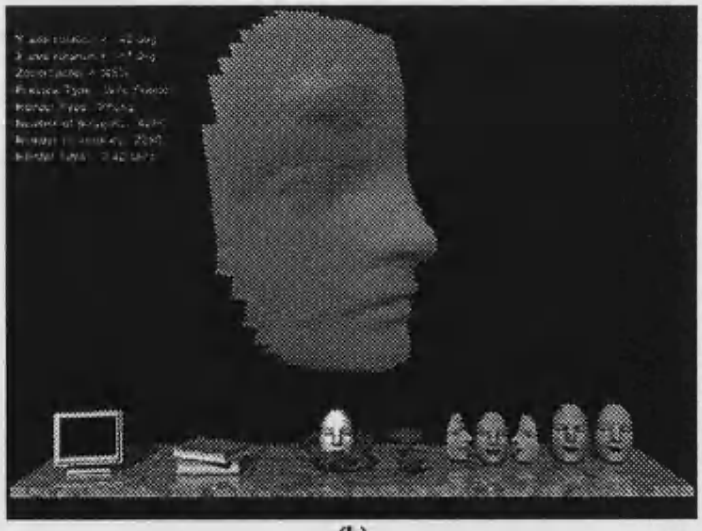
Table 7.4: Comparison of the measured and calculated values for depth and areas of the cavities on the face.

Two types of facial problem, that include both convex and concave areas have been evaluated and it has been shown that, it is possible to assess the shape of the surface in a mathematically meaningful manner. These approaches may help in the better planning of clinical applications.

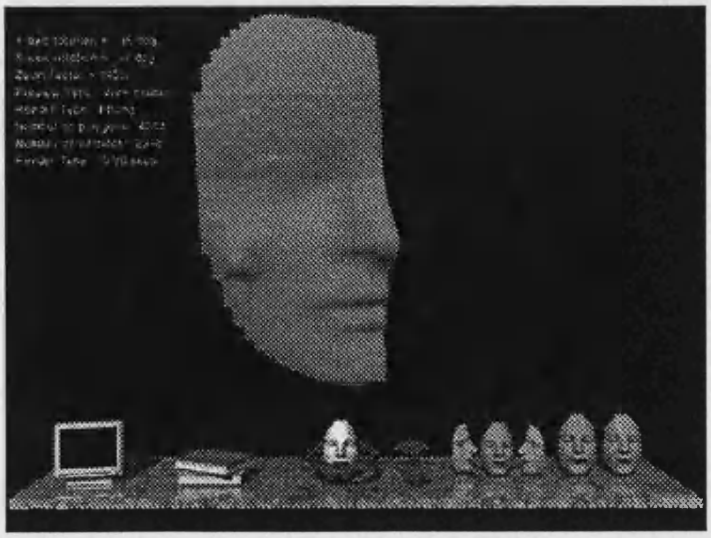
Other type of deformities on a face are caused by diversion or misalignment of the facial bones. Misalignment of the lower jaw, because of the mandible, and a diverted nose are examples for these types of deformities. Figure 7.14 and Figure 7.15 illustrates the reconstructed images from the deformed faces with the mentioned problems. The reconstructed images show that the system is useful to record the most common type of the facial deformities.



(a)

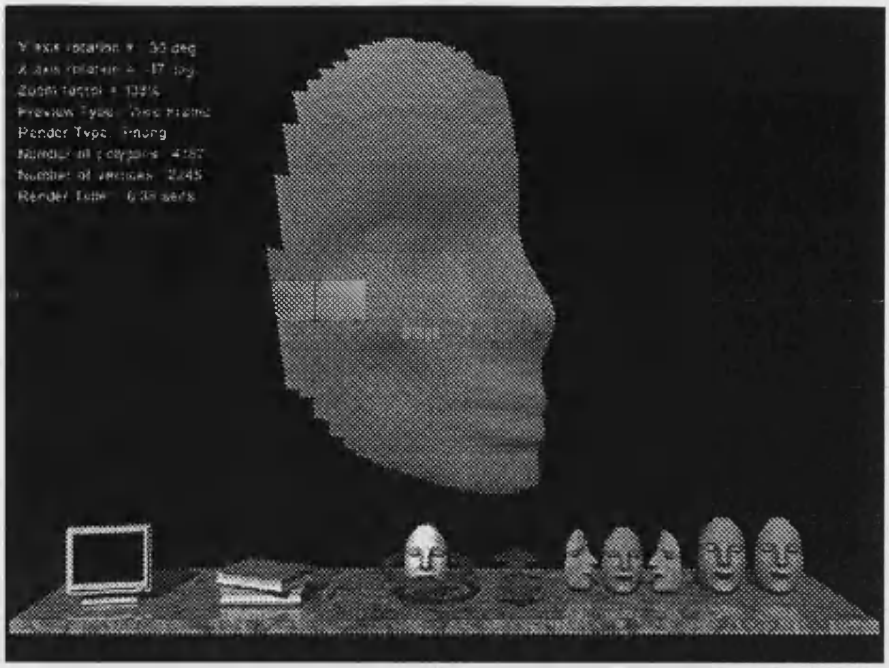


(b)

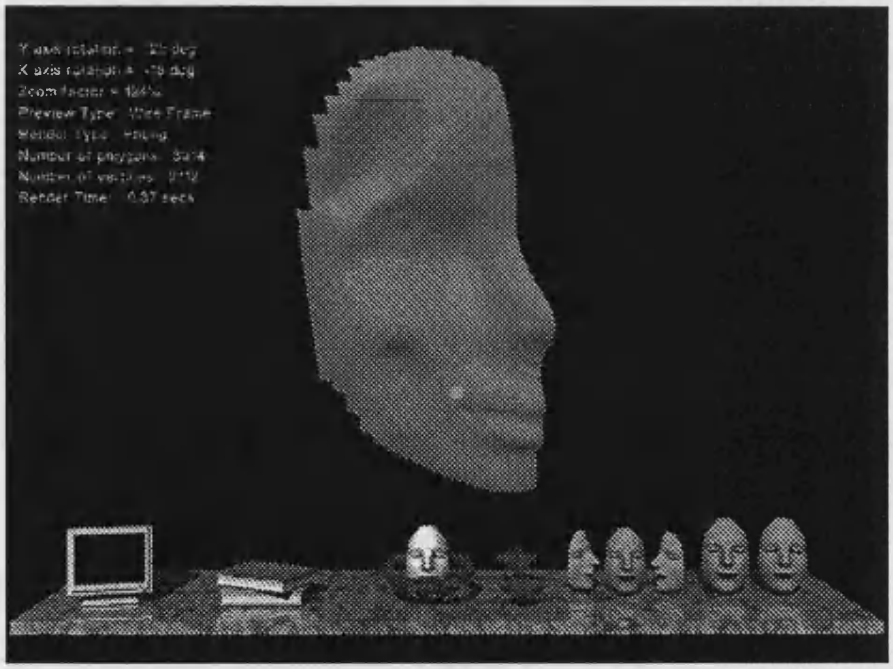


(c)

Figure 7.12: A lump on the (a) cheek near the mouth, (b) forehead and (c) cheek near the ear



(a)



(b)

Figure 7.13: A cavity on the (a) cheek near the mouth and (b) also eyebrow near the forehead

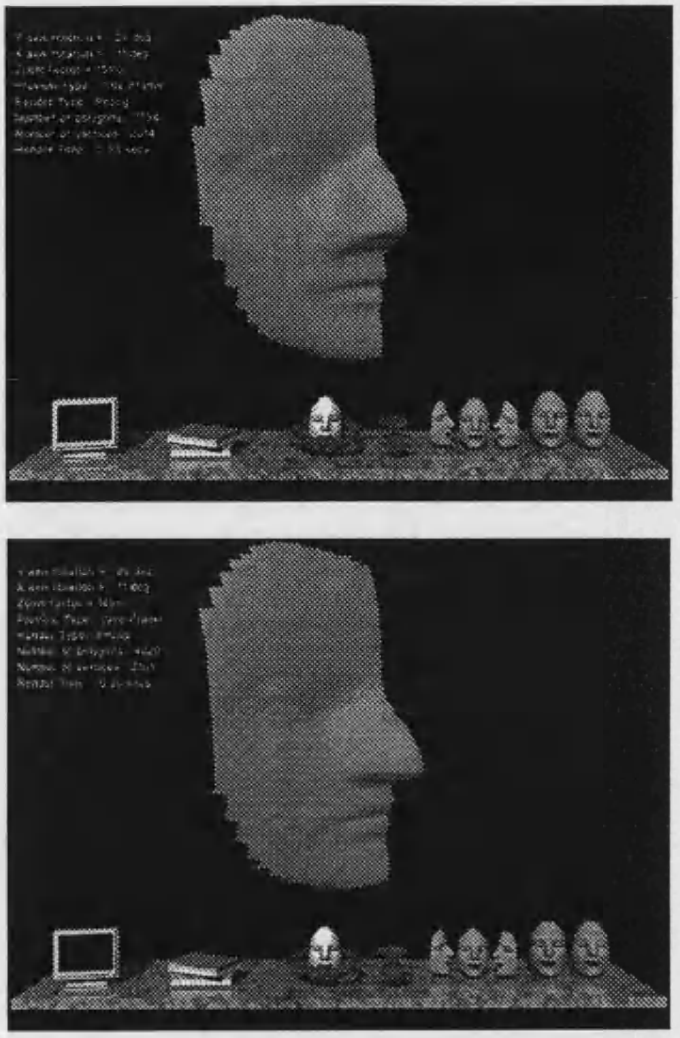


Figure 7.14: Diverted noses

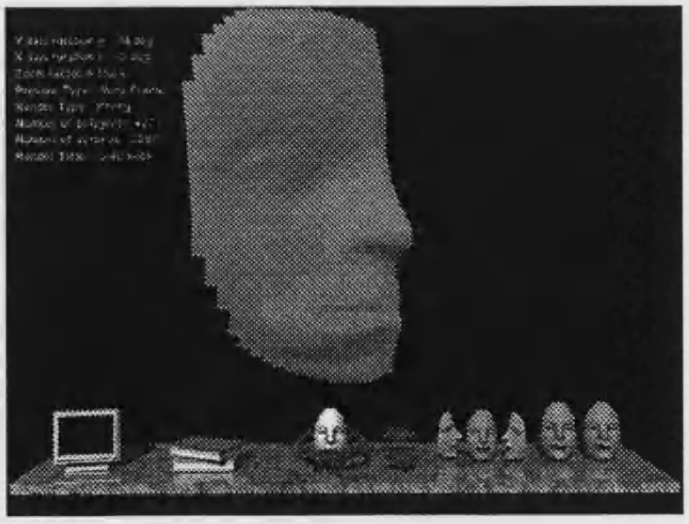


Figure 7.15: Deformity on the mouth and jaws.

The results from the reconstructed images satisfy quality of the implemented system in comparison with other range finding techniques.

If the performance criteria can be satisfied, measurement of body form has a multitude of applications. Notable examples include:

1. Facial: Deformities, surgery planning, surgery assessment, growth studies, objective abnormality description, expression.
2. Spinal deformity: Quantified description and assessment.
3. Lung function: Chest shape monitoring in breathing analysis.
4. Seating design: wheelchair users.

Chapter 8

Conclusions and Future Work

8.1 Conclusions

The measurement of facial soft tissue is fundamental to three-dimensional planning of cosmetic surgery procedures, growth studies and objective deformity analysis. The use of high resolution computer displays offers a very effective means of allowing analysis and further data extraction from the data base with a minimum of training and tailoring of systems to the exact requirements of the physician.

Established photogrammetric techniques offer high accuracy but with time consuming manual analysis. Laser scanning systems can acquire surface data at rates in excess of 700 points per second with measurement accuracy better than 0.5 mm, but these require the patient to remain still for several seconds. Phase analysis of Moiré fringes can utilise high speed image capture, but in both these techniques fusion of original patient images, surface data and anatomical landmarks can be difficult. CCD-based biostereometrics offers rapid image capture, which is particularly important in the measurement of young children or patients who are mentally confused or have impaired muscular control, and patient images that retain important landmarks for registration.

A complete system for non-invasive 3-D surface imaging has been developed capable of accurately recovering the 3-D shape of a portion of the human face. In the context of surface imaging of the human face, it has been shown how parallel structured light is suitable with a two-dimensional pattern, and freedom from difficult accuracy problems. The system has been used successfully on a PC to measure the human face or any static objects to an error standard deviation of 1 mm. It therefore achieves its goal of millimetre accuracy measurements.

A real-time frame store optimised for use in a clinical environment was developed. The frame store allows the acquisition of two frames from the cameras simultaneously with instant previewing so that images could be checked before downloading to a PC for processing. It was found that AGC's in the video camera made inappropriate adjustments and a manual gain control was used to control the grey levels. Also the hardware was designed to permit the user to control the optical set-up calibration by applying suitable signals to the actuators.

The optical system includes two CCD high resolution cameras, one for each side of a face, a controllable slide projector with enough power for illuminating different types of face and high resolution slides. The camera lens was chosen to cover a whole face without any distortion. The functions of the projector, such as changing the slides, focusing and the light intensity may be done automatically by the computer. The slides

were implemented by lithography-on-glass technique to prevent heating distortion and also to produce a high quality pattern on the subject's surface.

Image processing methods and algorithms have been developed for the efficient and accurate recovery of 3-D data as evidenced in the experimental results. The goal of the image processing stage is to generate structured light image without any discontinuity, by employing two main stages; feature extraction and feature interpretation. The image processing software that has been written for the auto-extraction stage includes enhancement, edge detection, thinning and pruning. Minimum hand intervention and processing time were considered when writing the algorithms to achieve real time and automatic algorithms.

The overall contrast of the images are already good due to adjusting the AGC manually in the frame-grabber from which the image was digitised. Preliminary image processing on the PC consisted of Gaussain filtering, to remove digitisation noise, and Canny edge detection followed by the hystersis thresholding as a way of binarising the lines.

A new line following strategy for thinning was developed to prepare a suitable skeleton. The thinning algorithm produced a one pixel wide skeleton of the edge detected image by generating the median of the grid stripes. The thinned lines obtained by this procedure have less discontinuities and the method is computationally efficient in comparison with other iteration methods. Obviously the last step of the classical image processing is pruning from the image any speckle noisy pixels and ensuring that each line has one pixel thickness. The implemented thinning methods and pruning have been applied on the edge detected images for comparing the results with the Canny edge detection. The result from the Canny edge detection is the most appropriate image for our application.

A key step in the 3-D reconstruction using structured light is to solve the grid labelling or matching problem, to find the correct correspondence between the detected grid points in the camera image and the points in the projected grid pattern. A method of matching for a structured light system is proposed by using a marked grid and applying the tracing algorithms. The special characteristic of the implemented algorithms are to correct all of the image problems first and then apply the labelling on a perfect image. This makes the labelling easy and robust in comparison with other implemented techniques based on propagating the defined constraints using the relaxation technique. The selected pattern gives a suitable sequence for the image components so that they can be used for tracing and labelling. Also it was shown that the designed pattern has enough accuracy for capturing and restoring the facial information and therefore extra patterns are not necessary.

The novel tracing algorithms, by applying the inverse difference coding and junction sequences, have been used to extract all of the necessary information from the closed contours on the image. The classification of the problems based on the defined sequences permits the algorithm to predict and correct the deficiencies accurately. The boundaries of the image, i.e. the reference line and boundary network, provide extra information for assessment of the loops during the loop tracing stage.

The most significant contribution is the use of the simple pattern to dramatically improve the grid labelling accuracy. When all the loops perfectly formed, the labelling algorithm can arrange and index the junctions on the vertical lines very easily. Preparing the structured light image by using the tracing algorithm and then applying the suitable labelling on the image components makes a robust processing for this stage. The labelling algorithm is based on line scanning applied on the processed image to index the junctions and vertical lines for the reconstruction stage.

The representation method was developed to prepare a realistic graphical image from a face. It became clear that there were special problems associated with the use of computer graphics for the reconstruction of a face which could not be solved by standard CAD techniques. The main differences arose from the amount and nature of the data to be handled and the speed with which results needed to be produced. The maximum number of pixels for reconstruction, by considering PC memory management and the quality of the reconstructed image, was selected and a high quality polygon mesh was built.

A user friendly 3-D computer graphics with a graphical icons environment was designed to generate a reconstructed model of the facial surface to allow the user to manipulate the face. The display method for the reconstructed image was improved by using high resolution graphics and a fast Phong rendering technique. The captured image from each side of a face should be reconstructed separately. Finally the joining algorithm was developed to join the sides of the face, by identifying the key features. This novel algorithms produced the results with necessary accuracy and finally ways of manipulating the deformities were reviewed.

The system was calibrated using a two stage technique, first calibrating the camera using a non linear camera calibration technique which includes radial distortion, then calibrating the projector using the results of the camera calibration. The principal advantages of the two-stage technique used are that it uses existing accurate camera calibration techniques and can easily accommodate new advances in the rapidly evolving field of camera calibration. Unlike the automatic calibration results reported by Tsai [117], the thin lens model does not introduce significant distortion in the 3-D measurement.

Feasibility of the imaging system has been demonstrated by examples illustrating the recovery of the shape of the facial surface with different type of deformities. Height, distance and area of the selected surface's patch were measured and compared. The accuracy of measurement is dependent on the accuracy to which the 3-D surfaces can be aligned. These results show that it is feasible to use the implemented system to recover the shape of the human face with an acceptable accuracy.

8.2 Future Work

The aim of this work has been to provide an automatic technique for generating a 3-D model of a human face from a structured light base system. However, it must be remembered that the reason we want to do this is to diagnose and monitor facial deformities. So the next stage is to use the 3-D data set to yield the medical applications. To do this in the present work would have been premature, as although the representation obtained looks realistic, a more through scientific examination and evaluation is needed.

The 3-D graphics environment allows the user to evaluate the reconstructed face but it is not a real tool for a surgeon. This needs to be developed in conjunction with the requirements of a surgeon. A system for planning facial surgery must be expected to provide a number of essential facilities. The surgeon should be able to interact with the images which should be selectable from any viewpoint. Further because of the importance of the facial appearance to the patient, it is highly desirable to be able to produce a realistic prediction of the post surgical appearance.

For greater accuracy and a better representation of the face more polygons are required which implies that more junctions need to be projected onto the face. The thickness of the projected lines is the main problem for this limitation. The number of vertical lines is increased by reducing the distance between the projector and face, but the field of view for covering the whole face will be reduced. One solution for this problem is to use the structured light laser diode [125] with the power less than 1.0 mw (Class II).

To capture an image of the whole head, three cameras are required. Changing the system to capture from three cameras can readily be done by implementing the hardware as shown in Figure 8.1. The most important feature is that the program on the Lattice PLD does not have to be altered. For the steering, the spare outputs and inputs in the CPIN byte of the I/O controller can be used. As the ADC has three inputs and an analogue output, just one ADC and one synchronisation separator are needed.

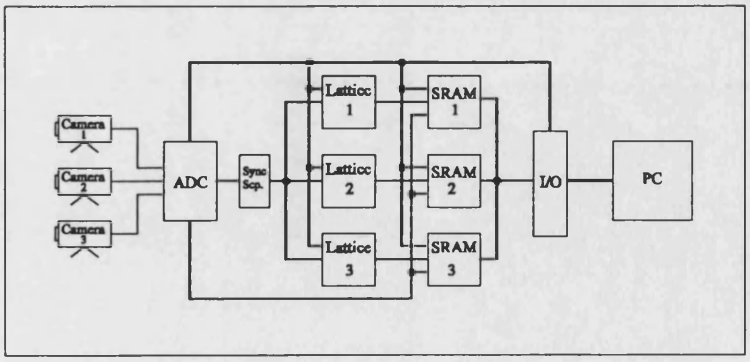


Figure 8.1: Set-up for capturing images from three cameras

Instead of using the I/O controller, a DMA controller can be used to get the data from the SRAMS to the PC RAM. This will increase the speed of the down loading time

because the data can directly be written to the memory without using the PC microprocessor and the normal bus procedure.

The prototype optical set-up could be improved by automating the calibration of the optic system, levelling the projector and adjusting the distances and angles. This can be done by implementing an optic system controller card to control the levelling and distance actuators. This card would be activated by the operator just by pressing a suitable button on the PC keyboard.

Appendix A: Optics for Machine Vision

In this appendix some useful concepts from optics are introduced [126 , 127]. At the first step the pinhole camera model is explained to determine the true focal length of the lens and then the maximum depth of field is calculated for preplanning of the experiments.

A.1 The pinhole camera model

The most suitable way of modelling a camera for calibration purposes is to use a pinhole model as shown in Figure A.1, where all rays of light are assumed to pass through the camera focal point and all image points are assumed to be in focus. This means that the camera focusing and apertures are ignored.

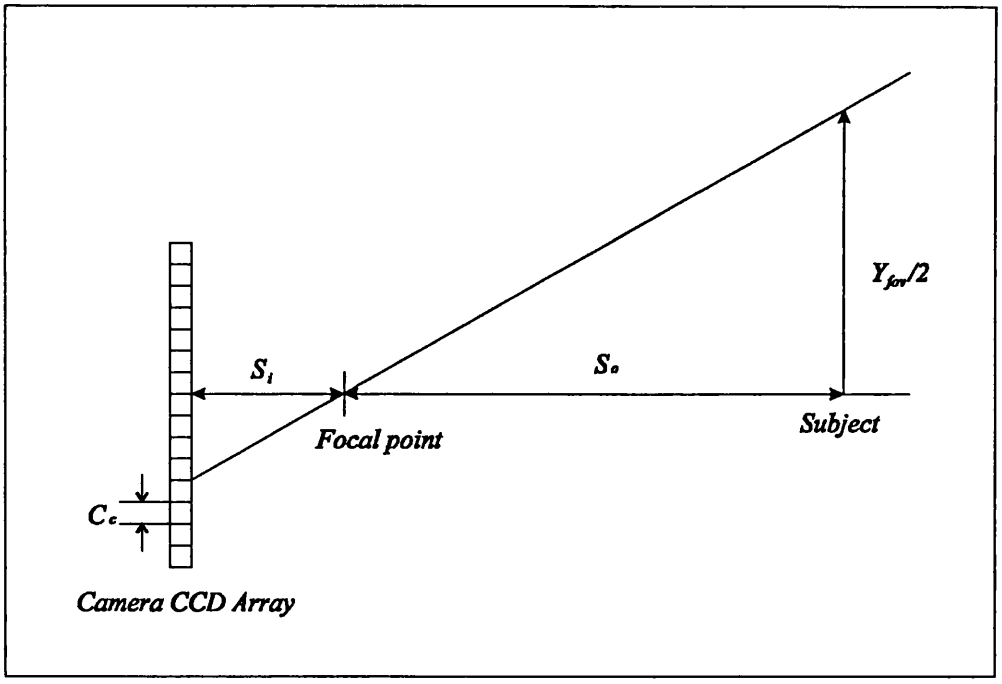


Figure A.1: 2-D illustration of the pinhole camera model. The camera has C_n pixels of size C_c. The distances S_i, S_o and the focal length f_c are related by the lens equation. Y_{fov} is the field of view in mm at the subject.

The lens equation is shown in three useful forms in equation A.1, describes the relationship between the camera focal length f_c and the distances S_i and S_o, which are the distances from the focal point to the object and the camera image plane respectively. Mapping this onto the problem of camera calibration, S_o corresponds to the distance from the focal point to the CCD array and is therefore equal to the camera focal length according to the pinhole model. From the equation we can see that if S_o tends to infinity the focal length will become equivalent to the focal length (i.e. $f_c=S_i$).

$$f_c = \frac{S_i S_o}{S_i + S_o}$$

$$S_i = \frac{f_c S_o}{f_c - S_o} \qquad \text{Equation A.1}$$

$$S_o = \frac{f_c S_i}{f_c - S_i}$$

We can use the above equations to convert our estimates of S_i generated by the camera or projector calibration to estimate of f_c using estimates of the camera to subject range S_o, also derived from the calibration process. This is useful since S_i is generally unmeasurable, and therefore unverifiable.

A.2 Depth of field

One of the most important limitations in structured light systems is depth of field. The combination of limited camera and projector focusing ranges is provoked by the fact that the camera and projector focal planes are at an angle to each other, so that towards the edges of the working volume one or the other will start to cause blurring. We can estimate the depth of field of the camera and projector by defining the maximum blurring that is acceptable, called the circle of confusion c_c, which for the camera equals the dimensions of a single CCD pixel. Where we realistically only aim to decode light stripes in the camera image with a minimum wavelength of four pixels, we can see that the equivalent circle of confusion for the projector is 1/4 of the stripe interval in mm. Both of these circles of confusion are therefore easy to determine.

For our camera (Hitachi KP-111K), the CCD pixel size is approximately 0.011 mm, and for the image 'model' we aimed to have around 40 stripes covering a maximum object size 100 mm, so that the circles of confusion for the camera and projector are (0.011) and (100/40/4=0.625) mm.

The depth of field consists of the sum of two sub-expressions, the front depth of field and the rear depth of field, as shown in equation A.2, and is expressed in terms of the circle of confusion c_c, the aperture D and the focal length f_c.

$$DOF = \frac{N_e c_c}{M^2} \cdot \frac{1}{\left(1 + \frac{S_o - f_c}{h}\right)} + \frac{N_e c_c}{M^2} \cdot \frac{1}{\left(1 + \frac{S_o - f_c}{h}\right)} = \frac{2 N_e c_c}{M^2 \left[1 - \left(\frac{S_o - f_c}{h}\right)\right]} \qquad \text{Equation A.2}$$

The following intermediary variables are defined:

- The hyperfocal distance h, which is the distance to the closest points that are in focus when the camera is focused at infinity, and is given by $h = f_c^2 \,/\, c_c N$.
- The magnification M, where $M = f_c \,/\, (S_o - f_c)$.
- The value N, where $D = f_c \,/\, N$.

- N corrected by the 'bellows factor' to give N_e, where $N_e = N(1+M)$.

Focal length f_c	4 mm
Circle of confusion c_c corresponding to 1 CCD pixel	0.011 mm
N	0.36
Hyperfocal distance $h = f_c^2 / c_c N$	4040 mm
S_o, the approximate range to the subject	310 mm
Magnification $M = f_c / (S_o - f_c)$	0.013
$N_e = N(1+M)$	0.364
Camera DOF corresponding to 1 CCD pixel	47.6 mm

(a)

Focal length f_p	90 mm
Vertical camera resolution	512
Circle of confusion c_c corresponding to 4 CCD pixels (35 mm slides)	35×4/512=0.273 mm
N	3.6
Hyperfocal distance $h = f_p^2 / c_c N$	8242 mm
S_o, the approximate range to the subject	700 mm
Magnification $M = f_p / (S_o - f_p)$	0.15
$N_e = N(1+M)$	4.131
Projector DOF	104 mm

(b)

Table A.1: Depth of field calculations for the optical set-ups corresponding to the test images, (a) the camera and (b) projector.

Table A.1 gives the result of depth of field calculation for the set ups corresponding to the test images using the above circles of confusion, and shows that the camera depth of field is the limiting factor. The calculations correspond well with observations, the camera depth of field is the limiting factor, and we conclude that this is a sensible way of planning to overcome depth of field limitations. It is however worth nothing that for relatively continuos shapes such as the human face it is possible to work over a somewhat greater range that indicated by the depth of field calculations by focusing on the narrowest stripe and relying on being able to use a larger circle of confusion and wider stripes.

Appendix B: Spline Curves

Spline Curves [128] are used to generate smooth, natural looking curves for the disconnected regions from a number of control points. There are two ways by which the curve can be constructed: approximation or interpolation. The difference is that with interpolation the curve passes through the control points whereas with approximation the curve needs only to pass near the control points, resulting in much simpler computation. Approximation will be considered first as it is relevant to both methods.

Instead of supplying an x-co-ordinate to a function and then calculating the y co-ordinate, spline curves are usually expressed in parametric form as shown in equation B.1,

$$P(\mu) = f(\mu, P_0, P_1, \cdots, P_N) \qquad \text{Equation B.1}$$

The $N+1$ control points, in vector notation, are denoted by P and the parameter μ is a continues variable which represents the locus of the spline curve. It is normalised and therefore $\mu=0$ and $\mu=1$ represent the curve at P_0 and P_N respectively.

The function f is chosen to give the desired response in between and at the end of points. Two important functions will now be looked at: Bezier curves and B-spline curves.

B.1 Bezier Curves

The general iterative form of the Bezier curve is shown in equation B.2,

$$P(\mu) = \sum_{i=0}^{N} P_i W(N, i, \mu) \qquad \text{Equation B.2}$$

where μ is a continues variable describing the locus of the curve, N is the number of knots and W is the Bernstein blending function.

The blending function determines how the knots 'pull' the curve and therefore the form of the curve between knots.

$$W(N, i, \mu) = \frac{N!}{i!(N-i)!} \mu^i (1-\mu)^{N-i} \qquad \text{Equation B.3}$$

Setting values for the start and end of the curve gives,

$\mu=0$ $\quad P(0)=P_0$
$\mu=1$ $\quad P(1)=P_N$

This exactly the desired result. However, for other values of μ, $P(\mu)$ will be a blend of all other P_i and lies the problem of Bezier curves, lack of locality. The effect of the curve is to smooth out detail, particularly when the knots are set out in a 'zigzag' way.

B.2 Cubic B-spline Curves

The B-spline curve is similar to the Bezier curve except for the blending function used. The blending function takes into account only the nearest knots, at most four, and so, compared to Bezier curves, the locality is much better as shown in Figure B.1.

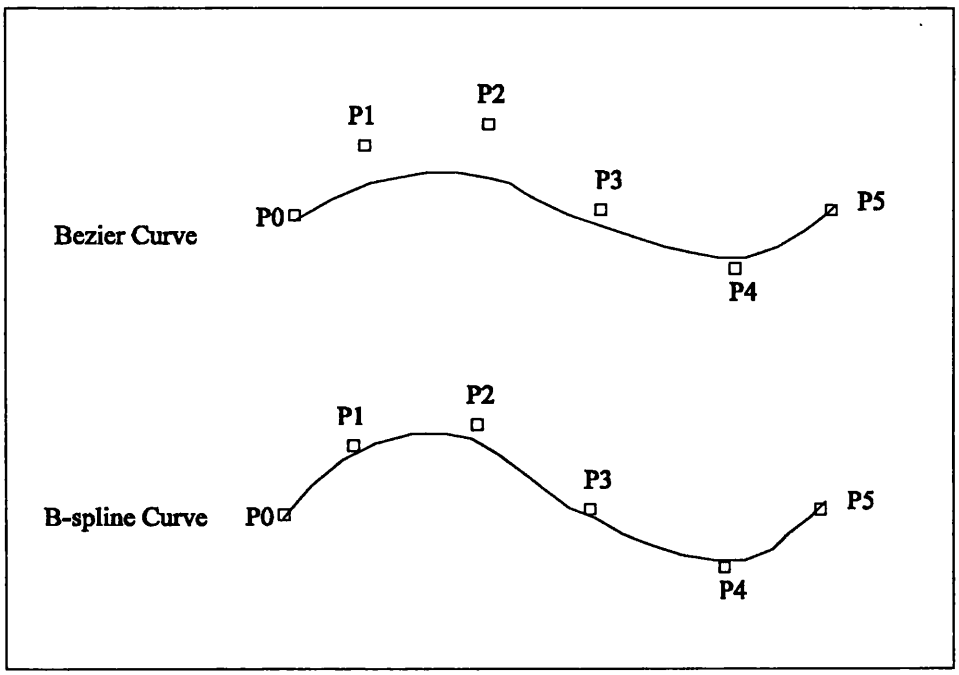


Figure B.1: Bezier and B-spline curves using the same control points

The blending function is shown in Figure B.2 and can be approximated piece wise as shown in equation B.4,

$$B(\tau)=\begin{cases}\frac{1}{6}(2+\tau)^3 & -2\le\tau\le-1\\ \frac{1}{6}(4-6\tau^2-3\tau^3) & -1\le\tau\le 0\\ \frac{1}{6}(4-6\tau^2+3\tau^3) & 0\le\tau\le 1\\ \frac{1}{6}(2-\tau)^3 & 1\le\tau\le 2\\ 0 & 2\le|\tau|\end{cases} \qquad \text{Equation B.4}$$

The blending function is convoluted with the control points,

$$P(\mu) = \sum_{i=-1}^{N+1} P_i B(i - N\mu)$$ Equation B.5

where $P_{-1} = P_0$ and $P_{N+1} = P_N$

Computing the convolution of equation B.5 for values of μ, ranging from 0 to 1, will produce the locus of the spline.

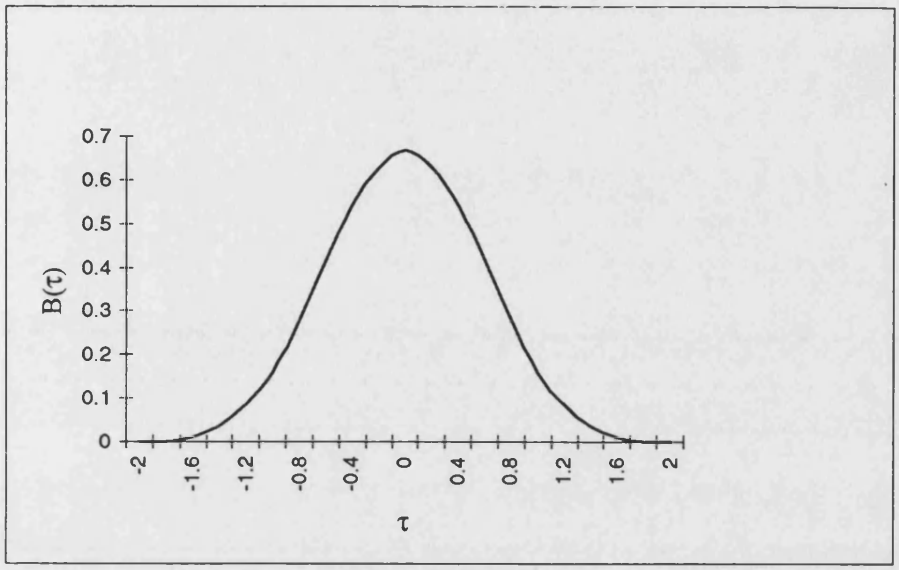


Figure B.2: B-spline blending function

The summation in the equation B.5 has been extended to include 'phantom' points on the start and end of the curve. Phantom points are needed to constrain the curve to pass through the end points P_0 and P_N.

B.3 B-spline Interpolation

It explained that how the curve passes near, but not through, the control points. To ensure the B-spline curve passes through the points P_i, parametric knots A_i is introduced,

$$P(\mu) = \sum_{i=-1}^{N+1} B(N\mu - i)A_i$$ Equation B.6

The parametric knots, A_i, are calculated from the real geometric knots, P_i, by using this fact that the former will be spaced at regular intervals of the parameter μ. Substituting $\mu = j/N$ in equation B.6 gives,

$$P\left(\frac{j}{N}\right) = P_j = \frac{1}{6}A_{j-1} + \frac{2}{3}A_j + \frac{1}{6}A_{j+1}$$ Equation B.7

Equation B.7 should be calculated for each geometric knot, but before doing this the end conditions must be considered. The clamped end condition will be used which

involves specifying end gradients. to estimate the gradient of any point on the spline, equation B.5 is differentiated with respect to μ,

$$P'(\mu) = N\sum_{i=-1}^{N+1} B'(N\mu - i)A_i \qquad \text{Equation B.8}$$

Equation B.8 involves differentiating the piece wise equation of equation B.4 and the result is shown in equation B.9,

$$B(\tau) = \begin{cases} \frac{1}{2}(2+\tau)^2 & -2 \le \tau \le -1 \\ \frac{1}{2}(-4\tau - 3\tau^2) & -1 \le \tau \le 0 \\ \frac{1}{2}(-4\tau + 3\tau^2) & 0 \le \tau \le 1 \\ \frac{1}{2}(2-\tau)^2 & 1 \le \tau \le 2 \\ 0 & 2 \le |\tau| \end{cases} \qquad \text{Equation B.9}$$

The gradient at the end points is found by substituting μ=*0* and μ=*1* into equation B.9 giving equations B.10 and B.11.

$$g_0 = P'(0) = N[A_{-1}B'(1) + A_0B'(0) + A_1B'(-1)] = \frac{N}{2}(A_1 - A_{-1}) \qquad \text{Equation B.10}$$

$$g_N = P'(1) = \frac{N}{2}(A_{N+1} - A_{N-1}) \qquad \text{Equation B.11}$$

These equations may be incorporated with equation B.7 into matrix form as shown in equation B.12.

$$\begin{bmatrix} \frac{N}{2} & 0 & \frac{N}{2} & 0 & 0 & \cdots & 0 & 0 & 0 \\ \frac{1}{6} & \frac{2}{3} & \frac{1}{6} & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} & 0 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & \cdots & \frac{2}{3} & \frac{1}{6} & 0 \\ 0 & 0 & 0 & 0 & 0 & \cdots & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} \\ 0 & 0 & 0 & 0 & 0 & \cdots & \frac{N}{2} & 0 & \frac{N}{2} \end{bmatrix} \cdot \begin{bmatrix} A_{-1} \\ A_0 \\ A_1 \\ \vdots \\ \vdots \\ \vdots \\ A_{N-1} \\ A_N \\ A_{N+1} \end{bmatrix} = \begin{bmatrix} g_0 \\ P_0 \\ P_1 \\ \vdots \\ \vdots \\ \vdots \\ P_{N-1} \\ P_N \\ g_N \end{bmatrix} \qquad \text{Equation B.12}$$

To calculate A_i from P_i, we need to do the inverse operation. If the matrix of equation B.12 is called *M*, then equation B.13 can be used to find *A*.

$$A = M^{-1} \cdot P \qquad \text{Equation B.13}$$

This equation may be solved using Gaussian elimination [129]. However, in the present case the curve can be made up of smaller segments and so a matrix of order 8 is perfectly adequate. The matrix inverse can then be simply pre-calculated and stored. When the values for A are calculated as shown in equation B.13, they may be used for calculating the spline in the usual way. An example matrix by considering only four reference pixels ($N=4$) and its inverse are shown in equation B.14.

$$M = \begin{bmatrix} -2 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1/6 & 2/3 & 1/6 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1/6 & 2/3 & 1/6 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1/6 & 2/3 & 1/6 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1/6 & 2/3 & 1/6 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1/6 & 2/3 & 1/6 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1/6 & 2/3 & 1/6 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1/6 & 2/3 & 1/6 \\ 0 & 0 & 0 & 0 & 0 & 0 & -2 & 0 & 2 \end{bmatrix}$$

$$M^{-1} = \begin{bmatrix} -0.539 & -0.464 & 1.856 & -0.497 & 0.133 & -0.036 & 0.01 & -0.003 & 2.137\times10^{-4} \\ 0.144 & 1.732 & -0.928 & 0.249 & -0.067 & 0.018 & -0.005 & 0.001 & -1.068\times10^{-4} \\ -0.039 & -0.464 & 1.856 & -0.497 & 0.133 & -0.036 & 0.01 & -0.003 & 2.137\times10^{-4} \\ 0.01 & 0.124 & -0.497 & 1.741 & -0.467 & 0.126 & -0.0036 & 0.009 & -7.479\times10^{-4} \\ -0.003 & -0.033 & 0.133 & -0.467 & 1.733 & -0.467 & 0.133 & -0.033 & 0.003 \\ 7.479\times10^{-4} & 0.009 & -0.036 & 0.126 & -0.467 & 1.741 & -0.497 & 0.124 & -0.01 \\ -2.137\times10^{-4} & -0.003 & 0.01 & -0.036 & 0.133 & -0.497 & 1.856 & -0.464 & 0.039 \\ 1.068\times10^{-4} & 0.001 & -0.005 & 0.018 & -0.067 & 0.249 & -0.928 & 1.732 & -0.144 \\ -2.137\times10^{-4} & -0.003 & 0.01 & -0.036 & 0.133 & -0.497 & 1.856 & -0.464 & 0.539 \end{bmatrix}$$

Equation B.14

Appendix C: Video Camera Singals and Hardware Programming

In this appendix, the additional information of the hardware includes of the video camera signals and Lattice-PLD programing are described in datails.

C.1 Video Camera Signals

The camera used for achieving the image is a charged coupled device (CCD) type giving a good low light response. The camera presents the picture at a BNC output. It gives out a start synchronisation pulse first, then line start synchronisation pulses each followed by an analogue voltage representing the luminance on a time basis associated to the x-position in the image.

C.1.1 Line synchronisation

At the beginning of each line the output of the camera is set to 0v for about 4.7μsec. This synchronisation signal can be recognised because it does not belong to the picture data. The range of picture data is between 0.3-1.0v, 0.3 for representing black and 1 for white. After this pulse the output of the camera will return to black level before the picture data is presented at the output.

The time scale of analogue output is matched by the x-position of the dots in the current line. At the end of the picture data the output is set to black level again and followed by another synchronisation signal for the next line as shown in Figure C.1.

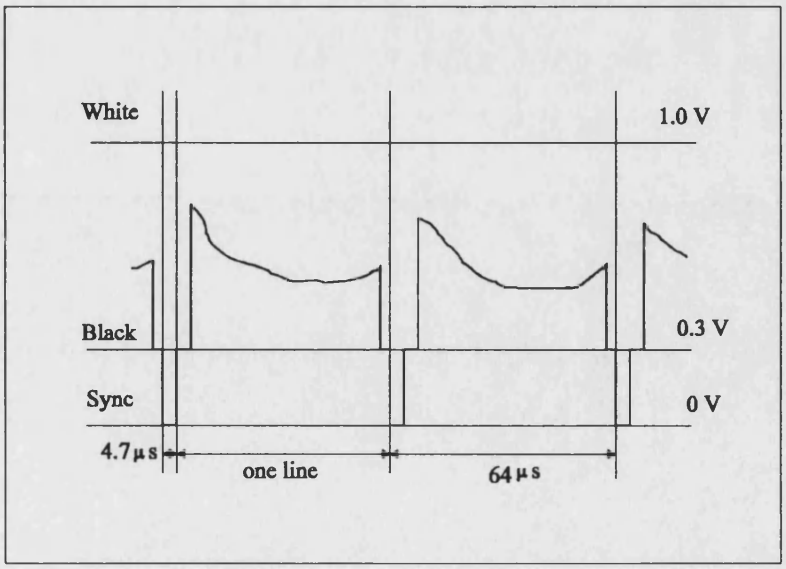


Figure C.1: Line synchronisation

C.1.2 Screen Synchronisation

Before line synchronisation pulse we need to adjust the screen with screen synchronisation pulse. This pulse is also at 0v level, but is longer than the line synchronisation pulse. This pulse is followed by a setting of black level which interrupted by line synchronisation pulses. After this pulse, 256 lines which be sampled in 20 msec give out as shown in Figure C.2. The next screen synchronisation is followed by the interlaced screen.

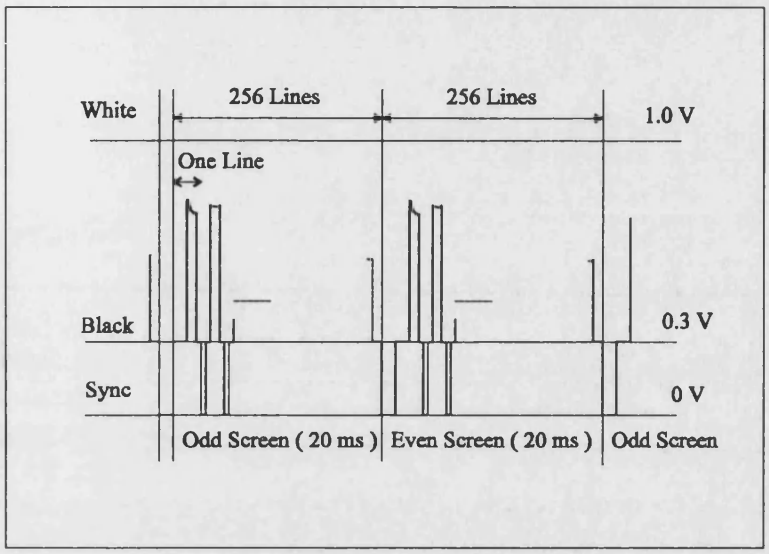


Figure C.2: Screen synchronisation

C.1.3 Odd and Even Screen

In order to ensure that the screen flickers at a rate high enough not to be noticeable to the human eye, each screen is divided into two parts as shown in Figure C.3. For a normal screen contains for example 512 lines with the total time of 40 msec will have a screen frequency of 25 Hz which causes twinkling of picture, therefore by dividing that into two screens with 256 lines a 20 msec for each is reached to a screen frequency of 50 Hz. So the lines of the screen with an odd number are assigned to the odd screen and are presented first as a full screen. This is followed by presenting the lines with even numbers of the same screen. As they have different positions on the screen and the human eye can see them long after they are presented, it seems like a full picture presented at 50 Hz. The next screen is output from the camera, also divided into odd and even screens.

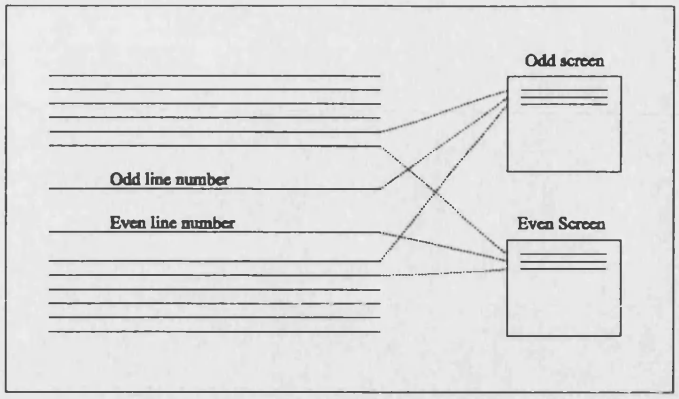


Figure C.3: Odd and even screen lines

C.2 Lattice PLD Programming

The Lattice ispLSI 1016 PLD contains 16 blocks, Generic Logic Blocks GLB, which the entered logic equations or macros can be assigned [130]. Those 16 generic logic blocks are connected by the router using the global routing pool. Each of these blocks contain special hardware. The detail about this hardware can be seen in the handbook contained in the developing system [62]. The 32 I/O cells can be connected to a physical I/O pin and can be programmed for different input or output types. Each of the two groups of 16 I/O cells share a common output routing pool to the pins. To get the incoming clock signal to the flipflops of the cells faster and with less skew than the normal data signals, a special clock distribution network on the Lattice is used.

The logic blocks on the chip are programmed from an integrated enviroment like a normal software compiler. The logic was either entered by putting in equations or by using macros for logic blocks, i.e. counters or flipflops. After the logic has been verified and routed to the given resources, the PLD is programmed by connecting the printer port of the PC to a plug mounted on the board with a cable supplied with the software package. The state machine is the main part of the PLD and determines the sequence of the processing. The states for digitising and downloading the image are explained below.

In state 0000, the state after power on reset or a previous grab and download, the PLD is waiting for the command to start the digitisation which is given by the PC. The signal used for that is the SAMPLE output of the PC from 0 to 1.

After that, the state machine changes to 0001 where it waits for the VSYNCH signal of the synchronisation separator indicating a new screen. When it given, the line and the pixel counters are set with RES_LINE_C and RES_PIX_C and the state machine switched to state 0111.

In this state, the pixel counter is used as a delay counter to count 122 (100 nsec $\times$ 122 = 12.2 µsec), by a CSYNCH signal. This delay is necessary to cover the black lines at

the beginning of each screen and to position the sampled lines into the middle of the lines given by the camera.

The next state is 0010, to wait for the beginning of the next line given by the next CSYNCH pulse from the synchronisation separator. When it is obtained, the state machine will be changed to 0011. After the beginning of the line, another delay is necessary because of the duration of the synchronisation pulse and the black level at the line, therefore the pixel counter was used as a delay counter again (10 nsec $\times$ 101 = 10.1 μsec).

In the state 0100 the digitisation is done. The pixel counter is incremented with the clock frequency until the end of the line given by the pixel counter overflow (EOL). During each incrementation one sample is made by putting the pixel count and the line count as an address to the SRAMs. The other signals that are needed, ADC_EN and WE signals are generated to open the output tri-state buffer of the ADC and SRAMs for writing. RAM_ACC chooses the appropriate SRAM for the odd or even screen given by the FRAME latch explained later.

After receiving EOL signal at the end of the processed line, state 0101 starts to find the last line, given by the overflow of the line counter (EOF). When this is not the case, the line number is incremented and the next line starts at the state of 0011.

At the end of the processed screen in the state 0110, a decision is made about grabbing the second screen. This is controlled by FRAME latch. When the flag is 0 another screen must be grabbed by writing for the VSYNCH pulse at the state 0001 again. But with a set FRAME flag, the screen has been completely grabbed, and the counters could be reset and the state machine is switched to state 1000 to wait for the command for downloading.

The download state machine is used for writing the data from the frame-grabber to the PC bus. The data during the download state is given in one stream, 16 bit output, includes of the two eight bit data with the same address from the SRAM's.

In the state 1000, the system is waiting for the sample command from the PC to change to 0 again, indicating the beginning of the downloading process. After that it switched to state 1001 to start the downloading process.

The next three states can be summarised as to alternate by writing the odd and the even bytes. In state 1001 the delay counter is reset for a delay explained later. In the following state 1010 the OE and RAM_ACC signals are given to write the output byte of SRAM1 to the lower byte of the I/O controller chip. After that, the frame will be set from 0 to 1 to write the even byte from SRAM2 in the same way to the higher byte of I/O controller. This is done in state 1011 before switching to state 1100.

In state 1100 a delay count is implemented to slow the digitiser card for matching with the slow PC bus. During this delay, the OE signal is extended through about half of the delay count without setting a RAM_ACC signal and is used as a synchronisation signal for the downloading process. After this delay, the FRAME will be reset to 0 to begin with the odd byte in the next cycle.

In the state 1101, the EOL signal, that is generated by the pixel counter overflow, is checked to write the next two bytes after incrementing the pixel counter in state 1111, if the line is not finished. Otherwise the state machine switches to state 1110.

In state 1110 the EOF is checked to go to the next line by incrementing the line count and resetting the pixel counter. At the end of the frame the downloading process is completed and the state machine is switched to state 0000 to wait for the next sample command to digitise the next screen.

The explained procedure has been done by applying the following program for the Lattice1016;

```
// FR_GRAB.ldf generated using Lattice 1016

LDF 1.00.00 DESIGNLDF;
DESIGN FR_GRAB;

PART ispLSI1016-90LJ;

OPTION MINIMIZE STRONGMIN;

OPTION PULLUP ALL;

OPTION Y1_AS_RESET OFF;

DECLARE

END; //DECLARE

SYM GLB  A2  1  ;
                                                        // PIXEL-COUNTER
CBU18( [ADDR0..ADDR7],ACARRY8,1,PIXEL_INC.PTCLK, RES_PIX_C);
END;

SYM GLB  A5  1  ;
                                                        // LINE-COUNTER
CBU18( [ADDR9..ADDR16], EOF, 1, LINE_INC.PTCLK, RES_LINE_C);
END;

SYM GLB  B0  1  ;
FD11(CLK, !CLK, CLK_20);            // DIVIDE 20MHZ-CLOCK TO 10MHZ
EQUATIONS
                                                        // PIXEL-COUNTER
PIXEL_INC = (!S3 & S2 & !S1 & !S0 & !EOL
        # !S3 & !S2 & S1 & S0 & !COUNT101) & CLK
        # S3 & S2 & S1 & S0
        # !S3 & S2 & S1 & S0 & !COUNT122 & !CSYNCH_;
                                                        // LINE-COUNTER
LINE_INC = (!S3 & S2 & !S1 & S0 & !EOF
        # S3 & S2 & S1 & !S0 & !EOF );
                                                        // DELAY-COUNTER
DELAY_CLK = S3 & S2 & !S1 & !S0 & !COUNT124 & CLK;
END
END;
```

```
SYM GLB  B2  1 ;
EQUATIONS
EN = !S3 & S2 & !S1 & !S0                          // ENABLE FOR RAM AND ADC
WE = EN;                                           // RAM-ACCESS
OE = S3 & !S2 & S1                                 // NEEDED OE FOR RAM
        # !COU6 & !COU5 & S3 & S2 & !S1 & !S0;
                                                   // PULSE FOR C-PRG-SYNC
ADC_EN = EN;                                       // ADC-ACCESS
COUNT101.RE = !S3 & !S2 & S1 & !S0;                // PIXEL-DELAY-FLIPFLOP
COUNT101.PTCLK = CLK;                              // WITH INPUT
COUNT101.D = ADDR6 & ADDR5 & !ADDR4 & !ADDR3 & ADDR2
         & !ADDR1 & ADDR0 & !S3 & !S2 & S1 & S0;
END
END;

SYM GLB  B3  1 ;
EQUATIONS
I_BPORCH_ = BPORCH_;                               // INVERTER - LINE
I_CSYNCH_ = CSYNCH_;                               // INVERTER - LINE
S2.PTCLK = CLK;                                    // STATE-MACHINE PART 1
S1.PTCLK = CLK;                                    // EQUATIONS KV-MINIMIZED
S2.RE = !POWER_ON_RESET_;                          // HARDWARE-RESET-INPUT
S1.RE = !POWER_ON_RESET_;
S2.D = !S3 & !S2 & S1 & S0 & COUNT101
   # !S3 & S2 & !S1 & S0 & EOF
   # S3 & !S2 & S1 & S0
   # S3 & S2 & !S1
   # S2 & !S1 & !S0
   # !S3 & S2 & S1 & S0 & !COUNT122
   # !S3 & !S2 & !S1 & S0 & !VSYNCH_;
S1.D = !S3 & !S2 & !S1 & S0 & !VSYNCH_
   # !S3 & !S2 & S1 & S0 & !COUNT101
   # !S2 & S1 & !S0
   # S3 & !S1 & S0
   # !S3 & S2 & S0;
END
END;

SYM GLB  B4  1 ;
EQUATIONS
FRAME.PTCLK = CLK;                                 // ODD/EVEN-FRAME-LATCH
FRAME.RE = FRAMERESET;                             // AND/OR NOT ALLOWED
FRAME.D =  !S3 & S2 & S1 & !S0                     // ELSEWHERE
        # S3 & !S2 & S1 & !S0
        # FRAME & (S2 # S1 # S0);
RES_PIX_C = !S3 & !S2 & S1 & !S0
          # !S3 & !S2 & S1 & S0 & COUNT101
          # !S3 & !S2 & !S1 & S0
          # !S3 & S2 & S1 & !S0
          # S3 & S2 & S1 & !S0 & !EOF
          # !POWER_ON_RESET_;
                                                   // LINE-DELAY WITHOUT FLIPFLOP
COUNT122 = !ADDR6 & ADDR5 & !ADDR4 & !ADDR3 & !ADDR2
        & !ADDR1 & !ADDR0;
END
END;

SYM GLB  B5  1 ;
```

```
EQUATIONS
RES_DEL_C = S3 & !S2 & !S1 & S0                    // RESET FOR DELAY-COUNTER
          # !POWER_ON_RESET_;
RES_LINE_C = !S3 & !S2 & !S1 & S0                  // RESET FOR LINE-COUNTER
           # !S3 & S2 & S1 & !S0
           # !POWER_ON_RESET_;
FRAMERESET = S3 & S2 & !S1& !S0                    // RESET FOR FRAME-FLIPFLOP
           # !POWER_ON_RESET_;
S3.RE = !POWER_ON_RESET_;                          // STATE-MACHINE PART 2
S3.PTCLK = CLK;
S3.D =  !S3 & S2 & S1 & !S0 & FRAME
    # S3 & !(S2 & S1 & !S0 & EOF);
END
END;

SYM GLB  B6  1 ;
EQUATIONS
ADDR8.CLK = PIXEL_INC;                             // PIXEL-COUNTER CONT RACE-FREE
ADDR8.RE = RES_PIX_C;
ADDR8.D = ADDR8 $ (ADDR7 & ADDR6 & ADDR5 & ADDR4 & ADDR3
                & ADDR2 & ADDR1 & ADDR0);
                                        // NO COU1, COU0 BECAUSE OF LIMITED INPUTS

COUNT124 = COU6 & COU5 & COU4 & COU3 & COU2;

                                                   // END OF LINE-SIGNAL
EOL = ADDR8 & ADDR7 & ADDR6 & ADDR5 & ADDR4 & ADDR3
  & ADDR2 & ADDR1 & ADDR0 ;
ADDR17 = !S3 & FIELD
    # S3 & FRAME;              // ODD/EVEN-LOGIC FOR ADDR17 FIELD INPUT
                               // FRAME JUST COUNTS WHILE SAMPLE,
                               // MAKES ADDR17 WHILE READING,CHOSES RAM
END
END;

SYM GLB  B7  1 ;
EQUATIONS
S0.RE = !POWER_ON_RESET_;                          // STATE-MACHINE PART 3
S0.PTCLK = CLK;
S0.D = !S3 & !S2 & !S1 & !S0 & SAMPLE
   # !S3 & !S2 & !S1 & S0
   # !S3 & !S2 & S1 & !S0 & !CSYNCH_
   # !S3 & S2 & !S1 & !S0 & EOL
   # S3 & !S2 & !S1 & !S0
   # S3 & S2 & !S1 & !S0 & COUNT124
   # S3 & S2 & !S1 & S0 & !EOL
   # S3 & S2 & S1 & !S0 & !EOF
   # !S3 & S2 & S1 & !S0 & !FRAME
   # !S3 & !S2 & S1 & S0 & !COUNT101
   # S3 & !S2 & S1 & !S0
   # S3 & S2 & S1 & S0
   # !S3 & S2 & S1 & S0 & !COUNT122;
END
END;

// No ldf text found for cell: A1

// No ldf text found for cell: A3
```

```
// No ldf text found for cell: A4

// No ldf text found for cell: A6

SYM GLB  B1  1 ;
FD11(NEG_CLK, !NEG_CLK, !CLK_20);                    // DIVIDE NEG 20 TO 10
EQUATIONS
CLK75NS = NEG_CLK # !CLK;                            // 75 NS RAM-PULSE
                                                     // RAM-ENABLE FOR BOTH
RAM_ACC = !S3 & S2 & !S1 & !S0 & !EOL & CLK75NS
                # S3 & !S2 & S1;
RAM1 = RAM_ACC & !ADDR17;                    // NOT INTERLEAVED , SRAM
RAM2 = RAM_ACC & ADDR17;                     // SEPARATES O/E TO CHIPS
END
END;

SYM GLB  A0  1 ;                             // MODIFIED DELAY COUNTER FOR LONGER
                                             // DELAY, MACRO NEEDS TOO MUCH ROOM
EQUATIONS

COU4.CLK = DELAY_CLK;
COU5.CLK = DELAY_CLK;
COU6.CLK = DELAY_CLK;

COU4.RE = RES_DEL_C;
COU5.RE = RES_DEL_C;
COU6.RE = RES_DEL_C;

COU4.D = COU4 $ (COU2 & COU1 & COU0 & COU3);
COU5.D = COU5 $ (COU2 & COU1 & COU0 & COU3 & COU4);
COU6.D = COU6 $ (COU2 & COU1 & COU0 & COU3 & COU4 & COU5);

END;
END;

SYM GLB  A7  1 ;                             // MOD. DELAY-COUNTER PART 2

EQUATIONS

COU0.CLK = DELAY_CLK;
COU1.CLK = DELAY_CLK;
COU2.CLK = DELAY_CLK;
COU3.CLK = DELAY_CLK;

COU0.RE = RES_DEL_C;
COU1.RE = RES_DEL_C;
COU2.RE = RES_DEL_C;
COU3.RE = RES_DEL_C;

COU0.D = !COU0;
COU1.D = COU1 $ COU0;
COU2.D = COU2 $ (COU0 & COU1);
COU3.D = COU3 $ (COU2 & COU1 & COU0);

END;
END;

SYM IOC  IO0  1 ;
```

```
XPIN IO _ADDR0                                  LOCK 25 ;
OB11(_ADDR0, ADDR0);
END;

SYM IOC IO1 1 ;
XPIN IO _ADDR1                                  LOCK 26 ;
OB11(_ADDR1, ADDR1);
END;

SYM IOC IO2 1 ;
XPIN IO _ADDR2                                  LOCK 27 ;
OB11(_ADDR2, ADDR2);
END;

SYM IOC IO3 1 ;
XPIN IO _ADDR3                                  LOCK 28 ;
OB11(_ADDR3, ADDR3);
END;

SYM IOC IO4 1 ;
XPIN IO _ADDR4                                  LOCK 29 ;
OB11(_ADDR4, ADDR4);
END;

SYM IOC IO5 1 ;
XPIN IO _ADDR5                                  LOCK 30 ;
OB11(_ADDR5, ADDR5);
END;

SYM IOC IO6 1 ;
XPIN IO _ADDR6                                  LOCK 31 ;
OB11(_ADDR6, ADDR6);
END;

SYM IOC IO7 1 ;
XPIN IO _ADDR7                                  LOCK 32 ;
OB11(_ADDR7, ADDR7);
END;

SYM IOC IO8 1 ;
XPIN IO _ADDR8                                  LOCK  4 ;
OB11(_ADDR8, ADDR8);
END;

SYM IOC IO9 1 ;
XPIN IO _ADDR9                                  LOCK 19 ;
OB11(_ADDR9, ADDR9);
END;

SYM IOC IO10 1 ;
XPIN IO _ADDR10                                 LOCK 20 ;
OB11(_ADDR10, ADDR10);
END;

SYM IOC IO11 1 ;
XPIN IO _ADDR11                                 LOCK 21 ;
OB11(_ADDR11, ADDR11);
END;
```

```
SYM IOC IO12 1 ;
XPIN IO _ADDR12                                    LOCK 22 ;
OB11(_ADDR12, ADDR12);
END;

SYM IOC IO13 1 ;
XPIN IO _ADDR13                                    LOCK 41 ;
OB11(_ADDR13, ADDR13);
END;

SYM IOC IO14 1 ;
XPIN IO _ADDR14                                    LOCK 8 ;
OB11(_ADDR14, ADDR14);
END;

SYM IOC IO15 1 ;
XPIN IO _ADDR15                                    LOCK 5 ;
OB11(_ADDR15, ADDR15);
END;

SYM IOC IO16 1 ;
XPIN IO _ADDR16                                    LOCK 6 ;
OB11(_ADDR16, ADDR16);
END;

SYM IOC IO22 1 ;
XPIN IO _RAM1_                                     LOCK 40 ;
OB21(_RAM1_, RAM1);
END;

SYM IOC IO23 1 ;
XPIN IO _RAM2_                                     LOCK 9 ;
OB21(_RAM2_, RAM2);
END;

SYM IOC IO24 1 ;
XPIN IO _OE_                                       LOCK 38 ;
OB21(_OE_, OE);
END;

SYM IOC IO25 1 ;
XPIN IO _WE_                                       LOCK 37 ;
OB21(_WE_, WE);
END;

SYM IOC IO26 1 ;
XPIN IO _ADC_EN_                                   LOCK 43 ;
OB21(_ADC_EN_, ADC_EN);
END;

SYM IOC IO27 1 ;
XPIN IO _SAMPLE                                    LOCK 3 ;
IB11(SAMPLE, _SAMPLE);
END;

SYM IOC IO28 1 ;
XPIN IO _CSYNCH                                    LOCK 18 ;
OB21(_CSYNCH, I_CSYNCH_);
```

```
END;

SYM IOC IO29 1 ;
XPIN IO _BPORCH                                    LOCK 17 ;
OB21(_BPORCH, I_BPORCH_);
END;

SYM IOC IO30 1 ;
XPIN IO _RESET_                                    LOCK 10 ;
IB11(POWER_ON_RESET_, _RESET_);
END;

SYM IOC I1 1 ;
XPIN I _VSYNCH_                                    LOCK 42 ;
IB11(VSYNCH_, _VSYNCH_);
END;

SYM IOC I2 1 ;
XPIN I _BPORCH_                                    LOCK 14 ;
IB11(BPORCH_, _BPORCH_);
END;

SYM IOC I3 1 ;
XPIN I _FIELD                                      LOCK 2 ;
IB11(FIELD, _FIELD);
END;

SYM IOC Y2 1 ;
XPIN CLK _CLK_20                                   LOCK 11 ;
IB11(CLK_20, _CLK_20);
END;

SYM IOC IO31 1 ;
XPIN IO _CLK                                       LOCK 39 ;
OB11(_CLK, CLK);
END;

SYM IOC IO17 1 ;
XPIN IO _S3                                        LOCK 15 ;
OB11(_S3, S3);
END;

SYM IOC I0 1 ;
XPIN I _CSYNCH_                                    LOCK 16 ;
IB11(CSYNCH_, _CSYNCH_);
END;

END; //LDF DESIGNLDF
```

References

[1] P. Waldhaus, G. Forkert, M. Rasse and B. Balogh, "Photogrammetric surveys of human faces for medical purposes", *SPIE, Close-Range Photogrammetry Meets Machine Vision*, vol. 1395, pp. 704-710, 1990.

[2] A. MacLeod, "Medical applications of close range photogrammetry", *Photogrammetric Record*, vol. 12, no. 68, pp. 155, 1986.

[3] H. Takasaki, "Moiré Topography", *Applied Optics*, vol. 9, no. 6, pp. 14671472, 1970.

[4] A. D. Linney, S. R. Grindrod, S. R. Arridge and J. P. Moss, "Three-dimensional visualisation of computerised topography and laser scan data for the simulation of maxilla-facial surgery", *Medical Informatics*, vol. 14, no. 2, pp. 109-121, 1989.

[5] M. Halioua and H. C. Liu, "Optical three dimensional sensing by phase measuring profilometry", *Optics and Laser in Engineering*, 11, pp. 185-215, 1989.

[6] E. A. Miskin, "The applications of photogrammetric techniques to medical problems", *Photogrammetric Record*, vol. 2, pp. 92, 1956.

[7] V. Sassoni, "Platoprint, physioprint, and roentgenographic, cephalometry, as new methods in human identification", *Journal of Forensic Sciences*, vol. 2, no. 4, pp. 428-442, 1957.

[8] J. R. T. Lewis and T. Sopwith, "Three-dimensional surface measurement by microcomputer", Butterworth & Co. (Publishers) Ltd., vol. 4, no. 3, pp. 159-166, 1986.

[9] A. T. Deacon, A. G. Anthony, S. N. Bhatia and J. P. Muller, "Evaluation of a CCD-based facial measurement system", *Medical Informatics*. vol. 16, no. 2, pp. 213-228, 1991.

[10] R. Y. Tsai, "A versatile camera calibration technique for high-accuracy 3-D machine vision metrology using off-the-shelf TV cameras and lenses", *IEEE Journals of Robotics and Automation*, vol. RA-3, no. 4, pp. 323-344, 1987.

[11] H. Grey, R. Warwick and P. L. Williams, *"Gray's Anatomy"*, 38th Edition, 1995.

[12] A. Silverstein, *"Human Anatomy and Physiology"*, John Wiley & Sons Inc., 1980.

[13] R. Ahlers and J. Lu, "Stereoscopic vision- an application oriented overview", *Int. Soc. Optical Eng.*, vol. 1194, pp. 298-308, 1989.

[14] P. Cavanagh, "Reconstructing the third dimension: interactions between colour, texture, motion, binocular disparity, and shape, *Computer Vision Graphics and Image Processing*, vol. 37, pp. 171-195, 1987.

[15] B. Gallup, C. Cotton, N. Shewchenko, S. Filoso and D. Hidson, "Evaluation of 3-D human shape monitoring techniques", *Surface Topography and Body deformity, Proceedings of Fifth International Symposium*, pp. 119-126, 1990.

[16] B. P. Gatliff and C. C. Snow, "From skull to visage'', *Journal of Biocommunication*, vol. 6, 27-30, 1979.

[17] W. H. Beeson, "Beeson facial plastic and reconstructive surgery", fps@beeson.com.

[18] A. M. Coombes, J. P. Moss, A. D. Linney, R. Richards and D. R. James, "A mathematical method for the comparison of three-dimensional changes in the facial surface", *European Journal of Orthodontics*, vol. 13, 95-110, 1991.

[19] A. M. Coombes, A. D. Linney, R. Richards and J. P. Moss, "A method for the analysis of the 3D shape of the face and changes in the shape brought about by facial surgery", *Proceedings of SPIE, Conference on Biostereometrics and Applications*, vol. 1380, 180-189, 1990.

[20] A. D. Linney, J. P. Moss, R. Richards, C. A. Mosse, S. R. Grindrod and A. M. Coombes, "Use of 3-D visualisation system in the planning and evaluation of facial surgery", *SPIE, Biostereometric Technology and Applications*, vol. 1380, pp. 190-199, 1990.

[21] S. T. Barnard and M. A. Fischler, "Computational stereo", *ACM Computing Surveys*, vol. 14, no. 4, pp. 553-572, 1982.

[22] Y. I. Abdel-Aziz and H. M. Karara, "Direct linear transformation from comparator co-ordinates into object space co-ordinates in close-range photogrammetry", *Proceedings of Symposium on Close-Range Photogrammetry*, Urbana, Illinois, 1971.

[23] P. Waldhaus, G. Forkert, M. Rasse and B. Balogh, "Photogrammetric surveys of human faces for medical purposes", *SPIE, Close-Range Photogrammetry Meets Machine Vision*, vol. 1395, pp. 704-710, 1990.

[24] L. Dorffner, G. Forkert, H. Klimpfinger and W. Rusch, "The application of visualisation techniques in facial surgery for diagnosis and quality control", *Proceedings of SPIE*, vol. 2357, pp. 194-200, 1994.

[25] A. Busboon and R. J. Scalkoff, "Direct surface parameter estimation using structured light: A predictor-corrector based approach", Image and Vision Computing, vol. 14, no. 5, pp. 311-321, 1996.

[26] A. Gregory and R. T. Lipczyski, "The three dimensional reconstruction of human facial images", *Proceedings of the 15th Conference of the IEEE Engineering in Medicine and Biology*, pp. 130-131, 1993.

[27] S. M. Dunn, R. L. Keizer and J. Yu, "Measuring the area and volume of the human body with structured light", *IEEE Transactions on System, Man., and Cybernetics*, vol. 19, no. 6, pp. 1350-1364, 1989.

[28] H. Jun, P. DeCosta and M. Dunn, "Body surface area measurement with structured light", IEEE, pp. 95-96, 1991.

[29] J. Le Moigne and A. M. Waxman, "Projected light grids for short range navigation of autonomous robots", *Proceedings of 7th International Conference of Pattern Recognition*, pp. 203-206, 1984.

[30] R. A. Jarvis, "A laser time-of-flight range scanner for robot vision", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-5, no. 5, pp. 505-512, 1983.

[31] W. Frobin and E. Hierholzer, "A sterephotogrammetric method for the measurement of body surfaces using a projected grid", *SPIE, Applications of Human Biostereometrics (NATO)*, vol. 166, pp. 39-44, 1978.

[32] M. Keefe and D. R. Riley, "Capturing facial surface information", *Photogrammetric Engineering and Remote Sensing*, vol. 52, no. 9, pp. 1539-1548, 1986.

[33] H. C. Joel, "The corpograph-A simple photographic approach to three-dimensional measurement", *Proceedings of Int. Society of Photogrammetric Symposium on Biostereometrics*, pp. 622-633, 1974.

[34] A. R. Turner and S. J. Derek Harris, "Measurement and analysis of human back surface shape", *Int. Archives of Photogrammetry and Remote Sensing*, pp. 355-362, 1986.

[35] I. Inokuchi, S.-I. Kawakami, M. Maeta and Y. Masuda, "Evaluation of facial palsy by moiré topography", *SPIE, Holography, Interferometry, and Optical Pattern Recognition in Biomedicine*, vol. 1429 , 39-45, 1991.

[36] M. Halioua, H. C. Liu, T. S. Bowins and J. K. Shih, "Automated topography of human forms by phase-measuring profilometry", Alberti/ Drerup/ Hierholzer: Surface Topography and Spinal Deformity VI. Gustav Fisher. Stuttgart, New York, pp. 6-15, 1992.

[37] M. Cheng, "Phase measuring profilometry using sinusoidal grating", *ASME, Symposium on Flexible Automation*, **2**, pp. 1089-1096, 1992.

[38] S. T. Young, S. W. Yip, H. C. Cheng and D. B. Shieh, "Three-dimensional surface digitiser for facial contour capture", *IEEE Engineering in Medicine and Biology*, pp. 125-128, 1994.

[39] A. R. Williams, "Light sectioning as a three-dimensional measurement system in medicine", *The Journal of photographic Science*, vol. 25, pp. 85-92, 1977.

[40] H. T. Hertzberg and F. P. Saul, "The Lang-US AF. Contourmeter, a new type of measuring instrument", *American Journal of Physics and Anthropol*, vol. 13 (2), pp. 382, 1955.

[41] A. F. Roche and J. W. G. Wignall, "The Cyrtographometer: a new instrument for recording contours", *American Journal of Physics and Anthropol*, vol. 20, pp. 521, 1962.

[42] J. Cobb, "A projected grid method for recording the shape of the human face", *Royal Aircraft Establishment, Technical Report*, Sep. 1971.

[43] E. J. Lovesey, "The development of a simple three-dimensional facial measuring technique", *Proceeding of Int. Society. Photogram. Symp. on Biostereometrics*, Washington DC, pp. 147, 1974.

[44] W. D. Leivesley, "The reliability of contour photography for facial measurements", *British Journal of Orthodontics*, vol. 10, pp. 34-37, 1983.

[45] C. Ozturk, S. Dubin, M. E. Schafer, W-U Shi and M-C Chou, "A new structured light method for 3-D wound measurement", *Proceedings of the Bioengineering Conference 2593 IEEE*, pp. 70-71, 1996.

[46] G. Sansoni, S. Corini, S. Lazzari, R. Rodella and F. Docchio, "3-D Imaging of surfaces for industrial applications: Integration of structured light projection, Grey code projection and projector-camera calibration for improved performance", *Proc. of SPIE*, vol. 2661, pp. 88-96, 1996.

[47] H.-H. Loh and M.-S. Lu, "SMD inspection using structured light", *IEEE Conference Proceedings on Industrial Electronics*, pp. 1076-1081, 1996.

[48] M. Asada, H. Ichikawa and S. Tsuji, "Determining of surface orientation by projecting a stripe pattern", *Proc. Int. Conf. on Pattern Recognition*, pp. 1162-1164, 1986.

[49] N. Shrikhande and G. Stockman, "Surface orientation from a projected grid", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 11, no. 6, pp. 650-655, 1989.

[50] R. L. Keizer, H. Jun and S. M. Dunn, "Structured light: Theory and practice and practice and practice", SPIE, vol. 1406, pp. 88-97, 1990.

[51] O. D. Faugeras and G. Toscani, "The calibration problem for stereo", *Proc. Computer Vision and Pattern Recognition*, pp. 15-20, 1986.

[52] J. Gerber, P. Kuhmstedt, R. Kowarschik, G. Notni and W. Schreiber, "3-D coordinate measuring system with structured light", *Proceedings of SPIE, Photomechanics*, vol. 2342, pp. 41-49, 1994.

[53] Robert J. Schalkoff, "Digital Image Processing and Computer Vision", John Wiley & Sons Inc., 1989.

[54] Z. Guangjun, and W. Hong, "Method of establishing general mathematical model of structured light 3-D vision", *Proceedings of SPIE*, vol. 2899, pp. 662-666, 1996.

[55] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, "Numerical recipes in C", Cambridge University Press, Second Edition, 1992.

[56] M. D. Altschuler, K. Bae, B.R. Altschuler, J. T. Dijak, L. A. Tamburino and B. Woolford, "Robot vision by encoded light beams", *Three-Dimensional Machine Vision*, Takeo Kanade, Ed. Norwell, MA: Kluwer Academic Publishers, 1987.

[57] Maxim Integrated Products, "Video Products", New releases, 1994.

[58] Elantec, "High performance analogue integrated circuits", 1992.

[59] Signetics Philips Semiconductors, "Video Data Handbook", 1991.

[60] Integrated Silicon Solution, Inc., "High speed CMOS static RAM", 1995.

[61] Lattice Semiconductor Corporation, "Lattice Databook", 1994.

[62] Lattice Semiconductor Corporation, "Lattice Handbook", 1994.

[63] National Semiconductor, "Linear Databook 3", 1989.

[64] IQD Limited, "Crystal Product Databook", 1994.

[65] M. Tooley, "PC-based Instrumentation and Control", Second Edition, , Butterworth-Heinemann, 1995.

[66] Howard Hutchings, "Interfacing with C", Butterworth-Heinemann, 1995.

[67] P. M. Narendra and R. C. Fitch, "Realtime adaptive contrast enhancement", *IEEE Trans. Pattern Anal. Mach. Intell.*, **PAMI-3**, no. 6, pp. 655-661, 1981.

[68] R. Hummel, D. Lowe, "Computational considerations in convolution and feature-extraction in images", *From Pixels to Features*, J. C. Simon, Elsevier Science Publishers BV, 1989.

[69] E. R. Davies, "Design of optimal Gaussian operators in small neighbourhoods", *Image and Vision Computing*, vol. 5, no. 3, 1987.

[70] G. A. Baxes, "Digital image processing, principles and applications", John Wiley and Sons Inc., 1994.

[71] M. Sonka, V. Hlavac, and R. Boyle, "Image processing, analysis and machine vision", International Thomson Computer Press, 1993.

[72] R. C. Gonzalez and R. E. Woods, "Digital Image Processing", Addison Wesley, 1993.

[73] J. Canny, "A computational approach to edge detection", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **PAMI-8**, no. 6, pp. 679-698, 1986.

[74] E. Davis, "Machine vision: Theory, algorithms and practicalities", Academic Press, 1990.

[75] C.Y.Suen, and P S. P. Wang, "Thinning methodologies for pattern recognition", World Scientific Publishing Co. Pte. Ltd., 1994.

[76] L. Lam, S.-W. Lee and C. Y. Suen, "Thinning methodologies-a comprehensive survey", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 9, pp. 869-885, 1992.

[77] T. Y. Zhang and C. Y Suen, "A fast parallel algorithm for thinning digital patterns", *Communication of the ACM*, vol. 27, no. 3, pp. 236-239, 1984.

[78] T. Pavlidis, "Algorithms for graphics and image processing", Rockville, MD: Computer Science, pp. 195-214, 1982.

[79] K. Kedem and D. Keret, "A fast algorithm for skeletonising lines by midline technique", *Proceedings of International Computer Science Conference*, pp. 731-735, 1988.

[80] I. S. N. Murthy and K. J. Udupa, "A search algorithm for skeletonisation of thick patterns", *Computer Graphics and Image Processing*, vol. 3, pp. 247-259, 1974.

[81] R. M. K. sinha, "A width-independent algorithm for character skeleton estimation", *Computer Vision Graphics and Image Processing*, vol. 40, pp. 388-397, 1987.

[82] J. D. Dessimoz, "Specialized edge-trackers for contour extraction and line-thinning", *Signal Processing*, vol. 2, no. 1, pp. 71-73, 1980.

[83] O. Baruch, "Line thinning by line following", *Pattern Recognition Letters*, vol. 8, pp. 271-276, 1988.

[84] Y. Shirai, "Recognition of polyhedrons with a range finder", *Artificial Intelligent*, vol. 4, no. 3, pp. 243-250, 1972.

[85] J. Batlle, E. Mouaddib and J. Salvi, "Recent progress in coded structured light as a technique to solve the correspondence problem: A survey", Pattern Recognition, vol. 31, no. 7, pp. 963-982, 1998.

[86] J. L. Posdamer and M. D. Altschuler, "Surface measurement by space-encoded projected beam systems", *Computer Graphics and Image Processing*, vol. 18, pp. 1-17, 1982.

[87] K. L. Boyer and A. C. Kak, "Color-encoded structured light for rapid active ranging", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-9, no. 1, pp. 14-28, 1987.

[88] P. Vuylsteke and A. Oosterlinck, "3-D perception with a single binary-coded illumination pattern", *SPIE, Optics, Illumination and Image Sensing for Machine Vision*, vol. 728, pp. 195-202, 1986.

[89] D. H. Ballard and C. M. Brown, "Computer Vision", Prentice-Hall Inc., 1982.

[90] G. Stockman and G. Hu, "Sensing 3-D surface patches using a projected grid", *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 602-607, 1986.

[91] R. A. Morano, C. Ozturk, R. Conn, S. Dubin, S. Zietz, and J. Nissanov, "Structured light using pseudorandom codes", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 3, pp. 322-327, 1998.

[92] R. J. Valkenburg and A. M. McIvor, "Accurate 3D measurement using a structured light system", *Image and Vision Computing, Elsevier*, vol. 16, pp. 99-110, 1998.

[93] G. Hu, and G. Stockman, "3-D surface solution using structured light and constraint propagation", *IEEE transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 4, pp. 390402, 1989.

[94] T. Nishimura and T. Fujimoto, "Fast contour line extraction algorithm with selective thresholding according to line continuity", *Systems and Computers in Japan*, vol. 25, no. 3, pp. 101-110, 1994.

[95] C. Stanek, N. Shrikhande and G. Hu, "Decision making strategies in contour tracing", *SPIE Imaging Technologies and Applications*, vol. 1778, pp. 247-257, 1992.

[96] J. E. Bresenham, "Algorithm for computer control of a digital plotter", *IBM System Journal*, vol. 4, no. 1, pp. 25-30, 1965.

[97] A. A. Mufti, "Elementary computer graphics", Reston Publishing Company, Inc., 1983.

[98] G. Hu, "Segmentation using range data and structured light", *SPIE Intelligent Robotics and Computer Vision*, vol. 1381, pp. 482-489, 1990.

[99] D. H. Ballard, and C. M. Brown, "Computer Vision", Prentice Hall, Inc., 1982.

[100] J. R. Parker, "Practical computer vision in C", John Wiley & Sons, Inc. 1994.

[101] D. Yu and H. Yan, "An efficient algorithm for smoothing, linearisation and detection of structural feature points of binary image contours", *Pattern Recognition*, vol. 30, no. 1, pp. 57-69, 1997.

[102] J. A. Noble, "Finding half boundaries and junctions in images", *Image and Vision Computing*, vol. 10, no. 4, pp. 219-232, 1992.

[103] L. Parida, D. Geiger, and R. Hummel, "Junctions: detection, classification, and reconstruction", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 7, pp. 687-698, 1998.

[104] P. Meer, C. A. Sher, and A. Rosenfeld, "The chain pyramid: Hierarchical contour processing", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 4, pp. 363-375, 1990.

[105] D. P. Dobkin, S. V. F. Levy, W. P. Thurston, and A. R. Wilks, "Contour tracing by piecewise linear approximations", *ACM Transactions on Graphics*, vol. 9, no. 4, pp. 389-423, 1990.

[106] B.-D. Chen and P. Siy, "Forward/backward contour tracing with feedback", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-9, no. 3, pp. 438-446, 1987.

[107] D. Geiger, A. Gupta, L. A. Costa, and J. Vlontzos, "Dynamic programming for detecting, tracking, and matching deformable contours", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 3, pp. 294-302, 1995.

[108] B. Bell, and L. F. Pau, "Contour tracing and corner detection in a logic programming environment", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 9, pp. 913-917, 1990.

[109] Y.-T. Liow, A contour tracing algorithm that preserves common boundaries between regions", *CVGIP Image Understanding*, vol. 53, no. 3, pp. 313-321, 1991.

[110] T.-L. Chia, Z. Chen, and C.J. Yueh, "A method for rectifying grid junctions in grid-coded images using cross ratio", *IEEE Transactions on Image Processing*, vol. 5, no. 8, pp. 1276-1281, 1996.

[111] M. Unser, A. Aldroubi, and M. Eden, "Fast B-Spline transforms for continuous image representation and interpolation", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 3, pp. 277-285, 1991.

[112] C. P. Hubbal, "3-D facial imaging", *B.Eng. thesis, Department of Electronic & Electrical Eng, University of Bath*, May 1998.

[113] F. S. Hill, JR., "Computer Graphics", Maxwell Macmillan International Editions, 1990.

[114] P. Burger and D. Gillies, "Interactive Computer Graphics", Addison-Wesley, 1989.

[115] A. Watt, "Fundamentals of three-dimensional computer graphics", Addison-Wesley Publishing Company, 1989.

[116] S. Harrington, "Computer Graphics, a programming approach", McGraw-Hill, Inc., Second Edition, 1988.

[117] R. Y. Tsai, "An efficient and accurate camera calibration techniques for 3-D machine vision", *IEEE Computer Vision and Pattern Recognition*, pp. 364-374, 1986.

[118] R. K. Lenz, and R. Y. Tsai, "Techniques for calibration of the scale factor and image centre for high accuracy 3-D machine metrology", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, pp. 713-720, 1988.

[119] S. Shih, Y. Hung and W. Lin, "Accurate linear technique for camera calibration considering lens distortion by solving an eigenvalue problem", *Optical Engineering*, vol. 32, no. 1, 1993.

[120] J. Weng, P. Cohen and M. Herniou, "Camera calibration with distortion models and accuracy evaluation", *IEEE, Pattern Analysis and Machine Intelligence*, vol. 14, no. 10, 1992.

[121] H. A. Beyer, T. Kersten and A. Streilein, "Metric accuracy performance of solid state camera systems", *SPIE Videometrics*, vol. 1820, pp. 103-110, 1992.

[122] R. L. Keizer and S. M. Dunn, "Marked grid labelling", *Proceeding CVPR 1989, IEEE conference on Computer Vision and Pattern Recognition*, San Diego, USA, pp. 612-617, 1989.

[123] J. T. Tou and R. C. Gonzalez, "Pattern recognition principles", Addison-Wesley Publishing Company, 1974.

[124] L. H. Bimean, "Survey of design considerations for 3-D imaging systems", *SPIE, Optics, Illustration and Image Sensing for Machine Vision III*, vol. 1005, pp. 138-144, 1988.

[125] Lasiris Inc., "Laser diode structured light products", Laser Lines Ltd, 1998.

[126] J. Blackwell and S. Thornton, "Mastering Optics, An applications guide to optical engineering", McGraw-Hill, 1996.

[127] T. P. Monks, J. N. Carter and C. H. Shadle, "A real-time 3-D digitisation system for speech research", *SPIE, Biomedical Image Processing and Three-Dimensional Microscopy*, vol. 1660, pp. 819-829, 1992.

[128] P. Burger and D. Gillies, "Interactive computer graphics", Addison-Wesley Publishing Company, 1989.

[129] R. Sedgewick, "Algorithms", Addison Wesley, 1988.

[130] Lattice Semiconductor Corporation, "pLSI/ispLSI Development System", 1994.

Published work

1. B. Shokouhi, D. M. Monro and B. G. Sherlock, "A cost-effective system for facial imaging and three dimensional reconstruction", *Proceedings of SPIE, Optical Society of America, Medical Imaging Conference, Image Display*, Feb. 1998.

2. B. Shokouhi and B. G. Sherlock, "Facial imaging and three dimensional reconstruction for medical applications", *The Third Annual Conference of Computer Society of Iran, Iran University of Science & Technology*, Dec. 1997.

A cost-effective system for facial imaging and three dimensional reconstruction

S. B. Shokouhi[a], D. M. Monro[a], and B. G. Sherlock[b]
[a]Electronic & Electrical Engineering, University of Bath,
Claverton Down, Bath BA2 7AY, England
[b]Department of Engineering Technology, University of North Carolina at Charlotte,
9201 University City Boulevard, Charlotte, NC 28223, USA

ABSTRACT

Three dimensional (3-D) images have recently received wide attention in applications involving medical treatment. Most current 3-D imaging methods focus on the internal organs of the body. However, several medical image applications such as plastic surgery, body deformities, rehabilitation, dental surgery and orthodontics, make use of the surface contours of the body. Several techniques are currently available for producing 3-D images of the body surface and most of the systems which implement these techniques are expensive, requiring complex equipment with highly trained operators. The research involves the development of a simple, low cost and non-invasive contour capturing method for facial surfaces. This is achieved using the structured light technique, employing a standard commercial slide projector, CCD camera and a frame-grabber card linked to a PC. Structured light has already been used for many applications, but only to a limited extent in the clinical environment. All current implementations involve extensive manual intervention by highly skilled operators and this has proven to be a serious hindrance to clinical acceptance of 3-D imaging. A primary objective of this work is to minimise the amount of manual intervention required, so that the system can be used by clinicians who do not have specialist training in the use of this equipment. The eventual aim is to provide a software assisted surgical procedure, which by merging the facial data, allows the manipulation of soft tissue and gives the facility to predict and monitor post-surgical appearance. The research focuses on how the images are obtained using the structured light optic system and the subsequent image processing of data to give a realistic 3-D image.

Keywords: structured light, image processing, optic system calibration, 3-D medical imaging

1. INTRODUCTION

The image capturing system discussed in this report provides automatic algorithms to produce three-dimensional reconstruction of the surface of the face from two dimensional captured images. Each side of the face is captured and processed separately and then combined together to produce a complete 3-D reconstructed image. The steps involved in producing the 3-D image are shown in Figure 1. Each step was designed and implemented with the necessary accuracy for a completely automatic reconstruction system.

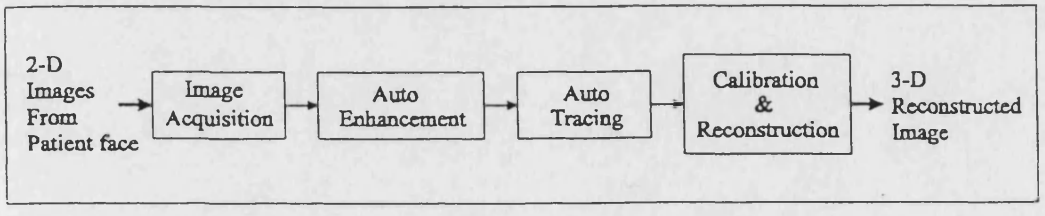


Figure 1: Stages of processing involved in image capture and 3-D reconstruction

Part of the SPIE Conference on Image Display • San Diego, California • February 1998
SPIE Vol. 3335 • 0277-786X/98/$10.00

2. DISCUSSION OF TECHNIQUES FOR FACIAL MEASUREMENT

There are many established techniques[1] for performing three dimensional measurements including; stereo photogrammetry, raster stereo photogrammetry, moiré topography, phase measuring profilometry and the laser scanning method. Before discussing the merits of each system, it is necessary to set out the characteristics of a desirable system and then use that as a reference against which the techniques will be judged. The optimum system should be portable, easy to set up, fast and economical. An accuracy of about one millimetre (or less) is appropriate to detect the required information from the face. Laser scanning methods[2] are the most accurate (less than 0.5 mm) of all techniques due to the ability of focusing a small powerful beam on the subject's face. It typically requires a data capture time of 10-15 second and any facial movement during this time causes inaccuracy on the three dimensional reconstruction. The alignment of the laser and camera is critical. The moiré fringe method[3] is considered a fairly inexpensive technique but has serious problems when applied to the human face. It requires a certain amount of manual intervention and also the accuracy of moiré technique is the lowest among the techniques considered. The method is normally restricted to back surfaces where the spatial resolution required is lower, in the order of 3-5 mm. After laser scanning, the most accurate method is phase measuring profilometry[4]. This has a measurement precision ranging from 0.5 mm for the human trunk to 0.1 mm for the face but its major disadvantage is that the grid must be moved, for which high precision equipment is required. Among the many range finding methods, structured light has been used widely because of its simplicity, speed, economy and safety. The accuracy is less than phase measuring profilometry, but it can be easily implemented with readily available equipment such as a conventional cameras and projectors. The necessary accuracy was achieved from this technique by adjusting the optic system to the optimum position, by using high accuracy grids and by writing advanced image processing algorithms.

3. STRUCTURED LIGHT METHOD

Structured light is "the process of illuminating an object with a specific light pattern and observing the lateral position of the image with a camera from a known angle". If a line of light is projected and viewed from one side, the distortions in the projected line can be translated into height variations along the line. The idea of the contour photography is not a new one, since the 1920s workers have attempted to use the camera for producing solid sculptures of human subjects. The original method of contour photography for medical use was invented by Sassoni[5] in 1957, he used a light sectioning technique for his "physioprint" which he hoped could be used in forensic applications to record three dimensional facial measurement. Roche *et al.*[6] in 1962 designed a "cyrtographometer" which they used to study growth in the female breast. Cobb[7] (1971) and Lovesey[8] (1972) produced equipment for calculating facial surface area and volume to help in the design of an air-crew oxygen mask. This technique was evaluated by Leivesley *et al.* [9] in 1983, for measuring facial deformities. Dunn *et al.* [10] in 1987 also applied this technique to the quantification of the area and volume of surface burns. Although the structured light technique has been used in several industrial applications for 3-D machine vision[11], its medical applications have been more limited because of the inaccuracies caused by the skin colour and movement. Following the pioneering papers, doctors, engineers and scientists have investigated the technique with the goal of achieving a fully automated system for 3-D imaging of the human body, but all of the designed systems required a significant amount of human intervention to digitise the captured image by hand.

4. OPTIC SYSTEM AND HARDWARE

The implemented system is illustrated in Figure 2. The Optic System Controller (OSC) controls the levelling and positioning of the projector and cameras, by defining reference points on the subject's face and projecting the reference grid (grid 1) on these points. The position of the optic system is then be corrected manually or automatically to achieve the best image from the face. A 35 mm slide projector with a 70-120 mm zoom lens projects a grid of parallel thin white lines on each side of the patient's face. Each line on the captured image is distorted by the facial topography. With a series of vertical lines one obtains a series of profiles at different positions across the face. Using knowledge of the geometry of the optical system a model is constructed of the facial surface in three dimensions by analysing the individual lines. In practice, it was found that regular photographic slides were not suitable because the dark parts of

the slide were insufficiently opaque and caused a poor contrast level in the image. This problem was solved by using a lithography-on-glass technique instead of slide photography. The new slides generate high quality images and also greatly reduces the problem of gaussian lighting on the surface. Two CCD cameras with 585(h) × 485(v) pixels resolution and precision zoom lenses were used.

A high resolution, standard interface frame-store hardware card was designed and built. This card is plugged into the PC directly and can grab a frame with 512 × 512 pixel resolution and 256 grey levels in real time. There are many frame-stores commercially available but the main reason for the development of this one is its specific intention for use in a clinical environment. Clinical use requires several images to be grabbed and previewed so that a suitable image with regard to equipment set-up and patient positioning is achieved. The frame-store has a hardware preview facility whereby an image can be viewed before grabbing and the intensity of the camera may be adjusted to compensate for the subject's face luminance.

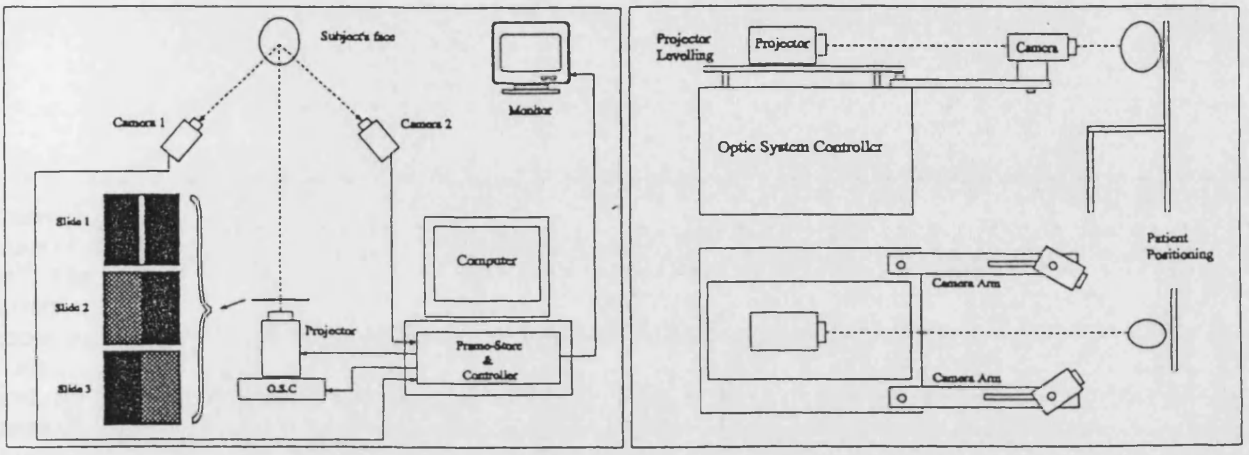


Figure 2 : Schematic diagram and layout of the designed system

5. IMAGE PROCESSING

The goal of the image processing stage is to generate facial contours without discontinuity and having single pixel thickness, by employing two main algorithms; auto enhancement and auto tracing. The use of white light as the structured light source rather than a laser avoids the risk of damage to the eye. However, because of the poorer contrast of the white light stripes, and because the projected white light can not be simultaneously in sharp focus over the entire surface, sophisticated image processing is necessary to produce a satisfactory image. Adaptive contrast enhancement is employed followed by Gaussian filtering. The enhanced image is then changed to a binary image by a directional edge detector. Vertical Sobel edge detection followed by thresholding produces a reasonable image. To reduce the lines to a single pixel width, a new algorithm was developed for line thinning which produces the median of the lines in a single pass. This method is computationally efficient and yields very good results with respect to both connectivity and contour noise immunity. Because of possible discontinuities in the lines an iterative method would cause additional loss of information.

After thinning the image, Speckle-filter algorithms remove noisy pixels from the resulting skeleton. In order to produce a 3-D data set, all of the lines must be perfectly formed, of single-pixel thickness and be continuous. The contours of the human face mean that certain parts of the projected grid can not be seen on the captured image (dead shadow areas) and breaks occur in the projected grid itself around the eyes, eyebrows and nose. To remove these defects, software has been designed to fill in the missing information by a tracing strategy as shown in Figure 3. Each line is defined with two reference pixels, the start and stop pixel. A tracing algorithm scans each line, pixel by pixel,

between the two reference pixels and during the scanning a filling algorithm corrects any discontinuities or overlapping by a decision making strategy[12].

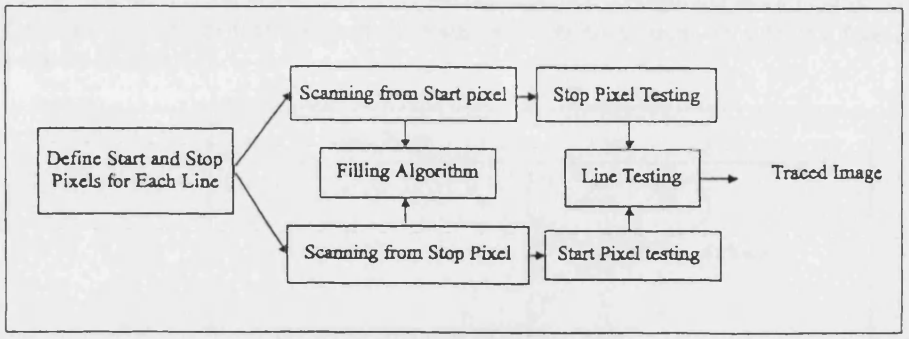


Figure 3 : Tracing algorithm stages

6. CALIBRATION

To achieve the necessary accuracy for medical applications, the system should be accurately calibrated. This involves correction of the metric inaccuracy of the solid state camera and projector[13] and the geometric calibration of the optical system[14]. The camera is calibrated first, by positioning a picture of squares in front of the camera and the camera parameters computed by calculating the distortion between the real picture and the captured image. The projector calibration matrix is determined next by positioning a flat screen in front of the camera and projecting the line pattern on it. After capturing the projected pattern the distance of the paper from projector is changed and a second image captured. By processing these images and using parallel plane projector model, the projection calibration matrix is computed. For the geometric calibration of the camera and projector, a series of targets or objects with known 3-D coordinates were used as shown in Figure 4. Development of an automatic calibration procedure for the camera and projector is currently being undertaken, along with detailed error and accuracy analysis for absolute height.

The position of the patient's head and the calibration of a reference line on the middle of nose is of vital importance and is done by projecting a reference line onto the face. In practice the patient should be positioned in a head restraining device, to enable accurate placement within the known geometric workspace, and should wear a black hat and necklace to eradicate the problems caused by the hair and neck.

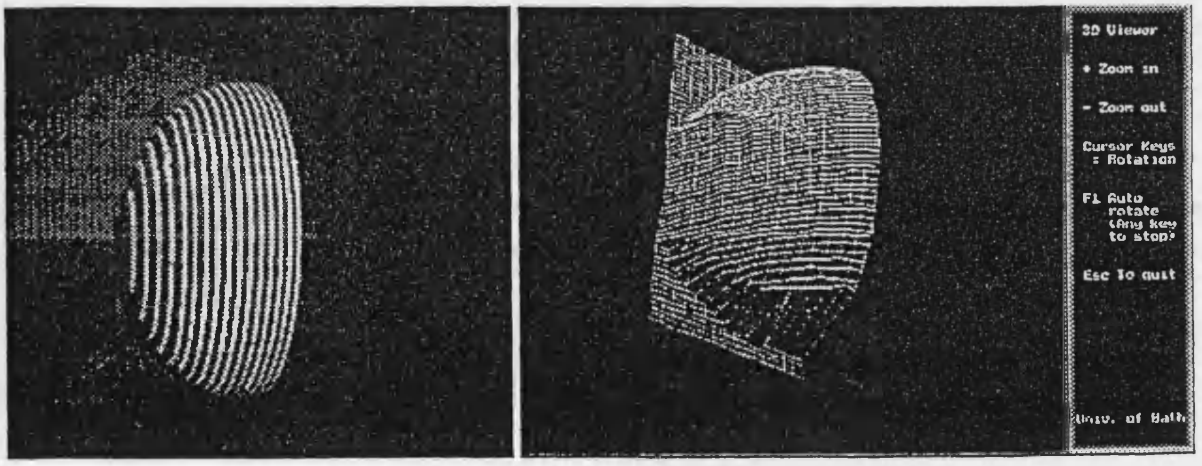


Figure 4: Spherical target and 3-D reconstructed image

7. THREE DIMENSIONAL RECONSTRUCTION

After an image has been enhanced to such a degree that the only information remaining is the array of projected lines, a three dimensional map can be obtained by defining a reference point on subject's face and then calculating the height as shown in Figure 5.

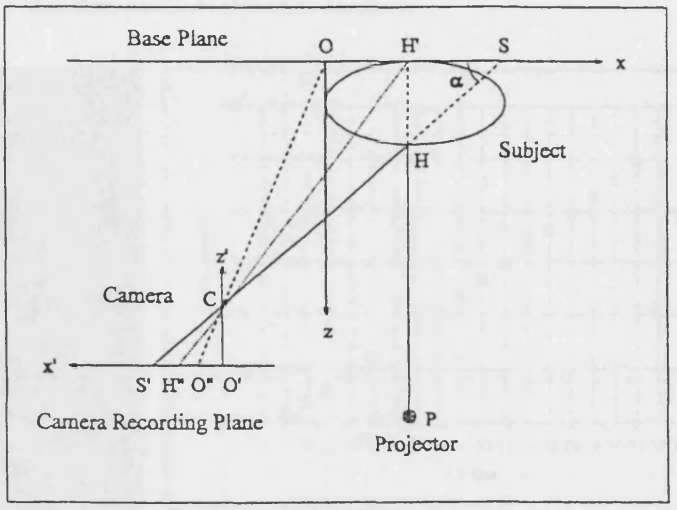


Figure 5 : Optic system geometry

Two co-ordinates are set in the system to calculate the locations of points on the subject's face. The first xyz co-ordinate is fixed to the patient positioning. The second co-ordinate is the transverse plane of the subject in the camera. By defining the reference point on the image to the centre of base co-ordinate (O) and using triangulation method, the height (HH') will be:

$$HH' = \frac{H'S \times d_2}{d + H'S} \tag{1}$$

Where d_2 is the distance of the camera from the base plane and d is the distance between the camera and projector. By applying the camera recording plane co-ordinate and considering the position of each line on the base plane, the equation (1) will change to the equation (2),

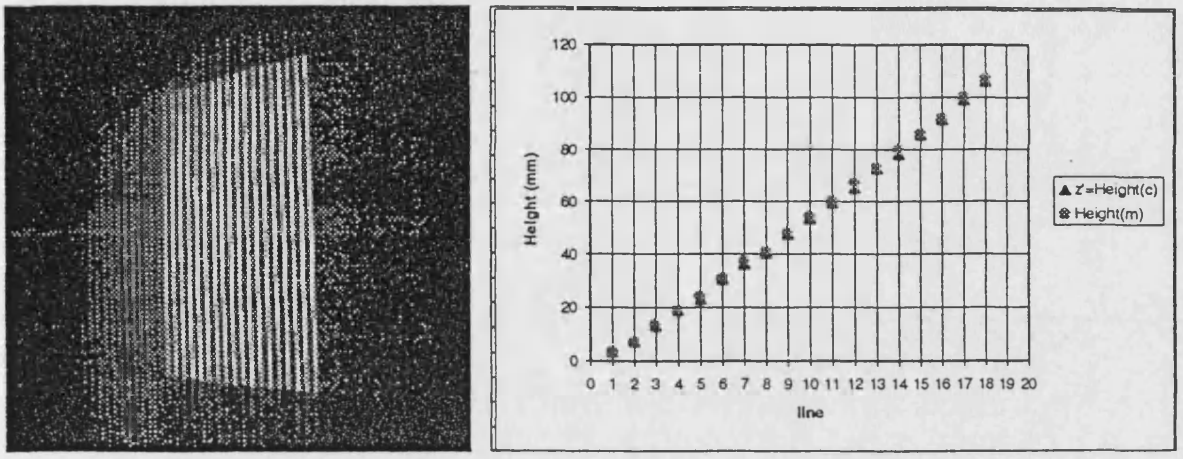
$$H'S(line)_1^N = \frac{d_2(x' \times line - x'_{ref})}{f_2 \times \sin(\alpha)} - l_o, \tag{2}$$

Where x' is the line position on the camera recording plane ($O'S'$), x'_{ref} is the reference point on the camera recording plane ($O'O''$), f_2 is the camera focal length and l_o is the distance between each projected line on the base plane.

By modelling the projector and having knowledge of its focal length f_1, the distance of the projector from the base plane d_1, and the distance between parallel lines on the slide l_s:

$$l_o(line)_1^N = l_s \times (1 + \frac{d_1}{f_1}) \times line, \tag{3}$$

According to the number of lines and resolution of the face, a grid with the same number of image lines is constructed. The processed image is then used, to insert the 3-D information into the grid. This is achieved by relating each of the labelled lines on the image to one of the vertical lines on the grid. Each individual line represents a slice through the subject in a different plane and it is first necessary to identify the individual lines and assign them each a value in the z-dimension. Each line is scanned in turn and its shape is related to the z-plane of the grid array. By combining equations (1), (2) and (3), the height equation can be obtained ($Height[line]_1^N$). This equqtion was evaluated by using a test object, a ramp at 45° to the base plane, as shown in Figure 6.



(a) (b)

Figure 6 : a) Projected grid on a flat object with 45° angle, b) Height accuracy, calculated and measured

The stages of the reconstruction algorithm are shown in Figure 7. After labelling and pixel continuity testing for each line, the geometric coefficients should be entered by operator for the height calculation. Additional data such as the calibration matrix is considered at this stage. By use of a certain amount of trigonometry, the captured image from two sides of the face is shown in Figure 8 and a 3-D reconstructed wire mesh model is formed as shown in Figure 9. This object can then be examined in three-dimensional space for medical applications. Also, by applying an array of triangular polygons instead of array of points for construction of the 3-D image, rendering the surface may be used to give shading effects (Figure 10).

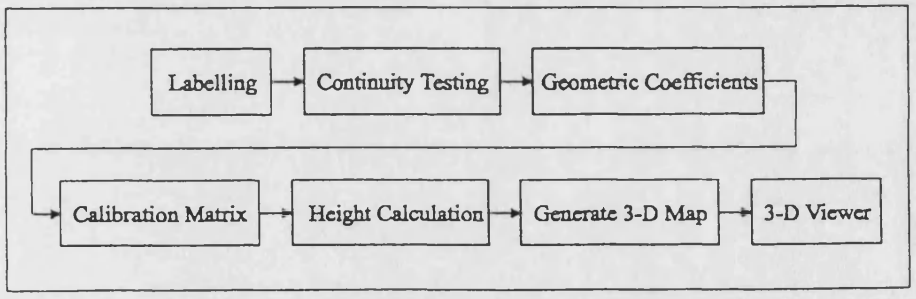


Figure 7 : Reconstruction stages

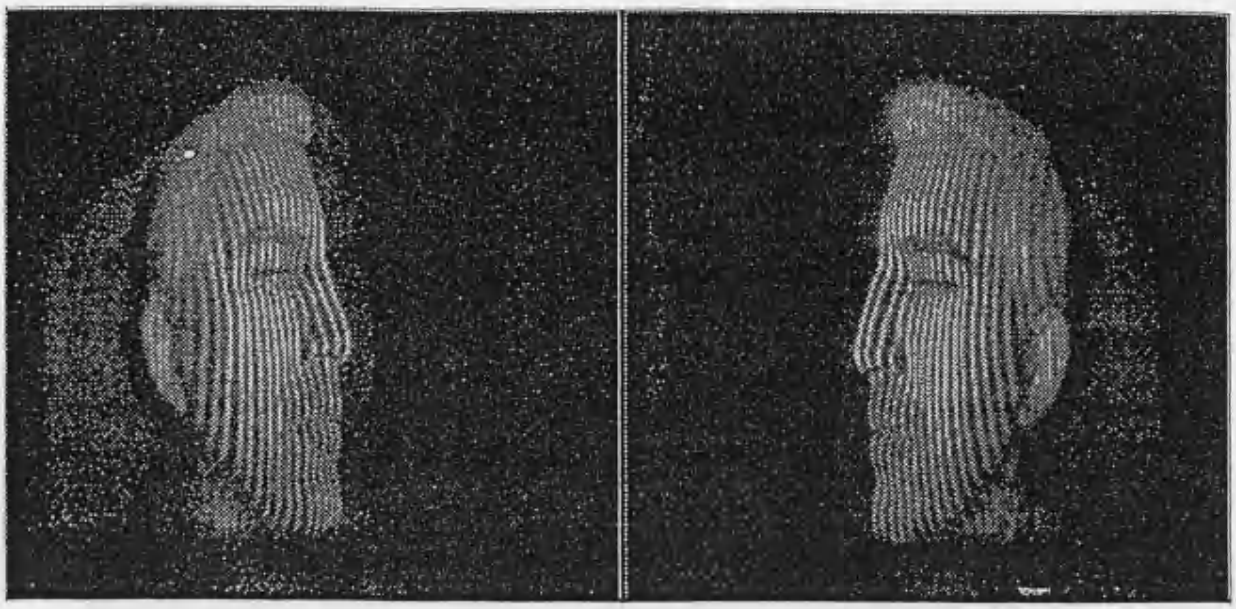


Figure 8 : Captured images from two sides of the face

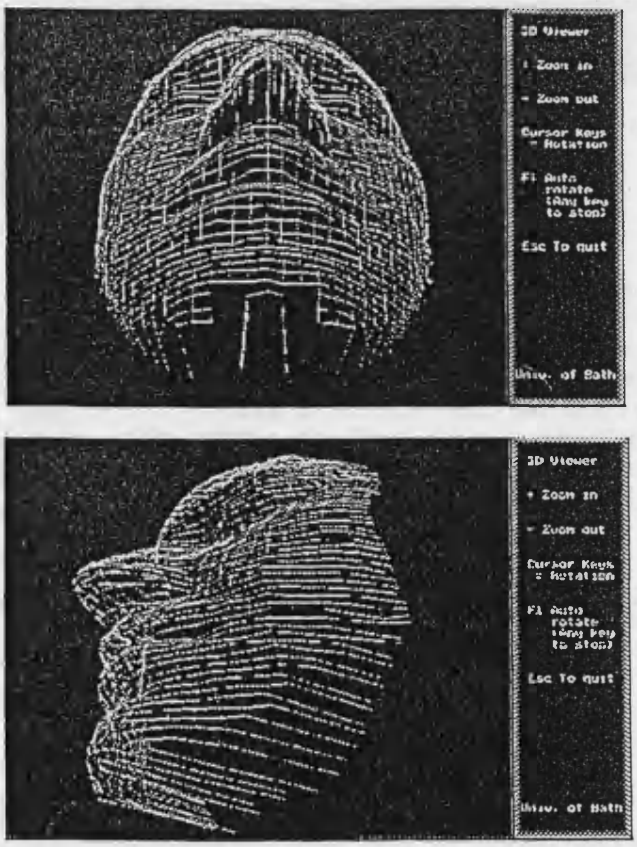


Figure 9 : Reconstructed wireframe images

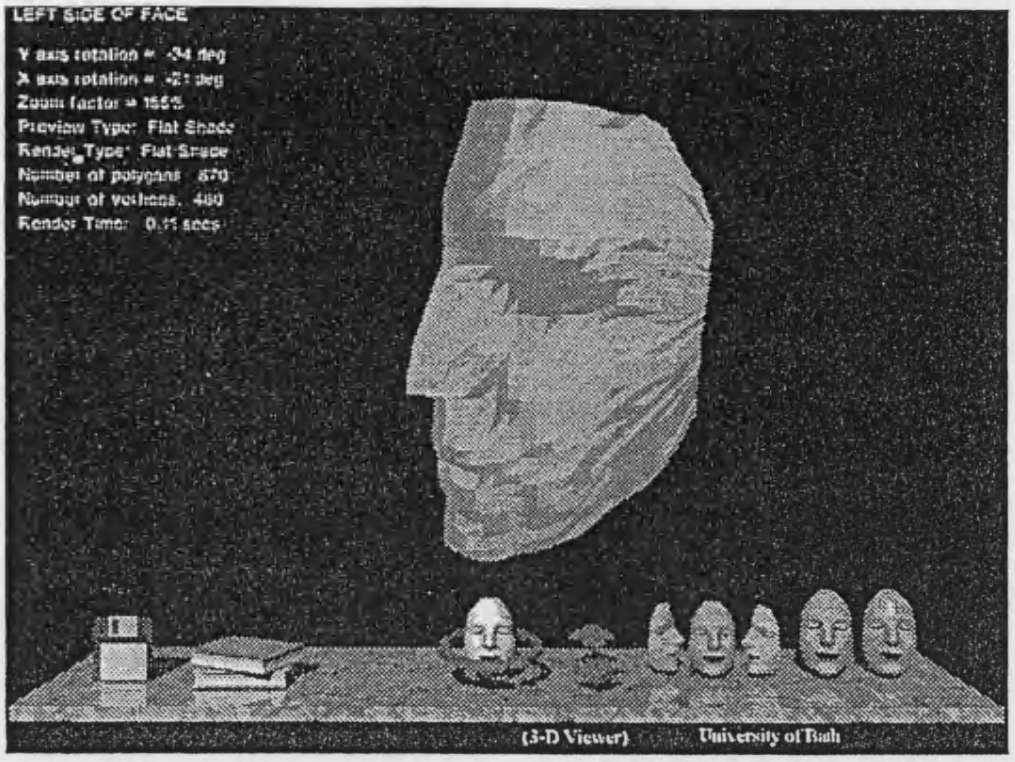


Figure 10 : The reconstructed shading face by a user friendly enviroment for clinical use

8. CONCLUSION

The implemented system and the reconstruction algorithms that have described perform well and have several advantages over other facial measurement systems. The system uses a non coherent light source, instant recording to eliminate errors due to the subject movement, simple optics and operation and links the hardware directly into the computer. By using photolithography-on-glass slides and CCD cameras, and by calibration and mathematical modelling of geometric errors, distortions are predictable and are corrected for. Image processing methods and algorithms have been developed for the efficient and accurate recovery of 3-D data. The software creates a user friendly operating environment which is suitable for operation by users with limited computer knowledge. The system is flexible and can also be applied to other body surfaces and hence a variety of other medical applications (Figure 11). Work is presently in progress in the following areas:

- Adapting the system to process injured faces by improving the optic accuracy and image processing.
- Developing the 3-D environment to incorporate the requirements of a surgeon. Incorporation of the surface along with X-ray data into CAD programs will give clinicians an extremely versatile tool for analysis of soft tissue and bone movement prior to surgery.

9. ACKNOWLEDGMENT

The idea for this work came out from the Royal hospital in Bath and I would like to thank the medical physicists and Dr. R. T. Lipczynski for their help and co-operation and also thanks from Dr. D. Ormondroyd and Dr. G. Allen for their great help.

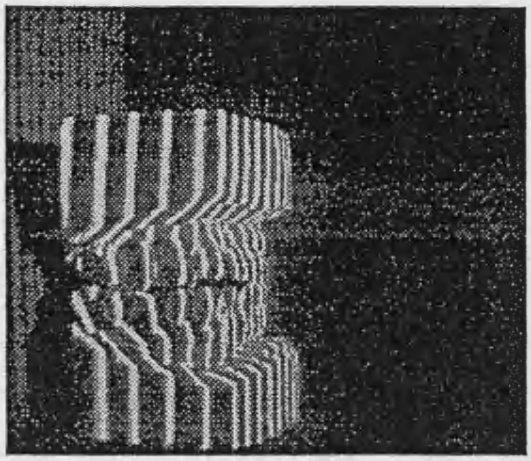


Figure 11 : Captured image from a dental cast

10. REFERENCES

1. B. Gallup, C. Cotton, N. Shewchnko, S. Filoso and D. Hidson, "Evaluation of 3-D human shape monitoring techniques", *Proceedings of Fifth International Symposium, Surface Topography and Body Deformity*, pp. 119-126, 1990.
2. A. D. Linney, J. P. Moss, R. Richards, C. A. Mosse, S. R. Grindrod and A. M. Coombes, "Use of 3-D Visualisation System in the Planning and Evaluation of Facial Surgery", *Proc. of SPIE, Biosterometric Technology and Applications*, **1380**, pp. 190-199, 1990.
3. I. Inokughi, S.-I. Kawakami, M. Maeta and Y. Masuda, "Evaluation of facial palsy by moiré topography", *Proc. of SPIE, Holography, Interferometry, and Optical Pattern Recognition in Biomedicine*, **1429**, pp. 39-45, 1991.
4. M. Halioua, H. C. Liu, T. S. Bowins and J. K. Shih, "Automated topography of human forms by phase-measuring profilometry", *Alberti/ Drerup/ Hierholzer: Surface Topography and Spinal Deformity VI. Gustav Fisher. Stuttgart. New York*, pp. 6-15, 1992.
5. V. Sassouni, "Palatoprint, physioprint and roentgenographic cephalometry, a new methods in human identification", *Journal of Forensic Sciences*, **2**, pp. 429-443, 1957.
6. A. F. Roche and J. W. G. Wignall, "The cyrtographometer: a new instrument for recording contours", *American Journal of Physics and Anthropology*, **20**, pp. 521, 1962.
7. J. Cobb, "A projected grid method for recording the shape of the human face", *Royal Aircraft Establishment, Technical Report*, Sep. 1971.
8. E. J. Lovesey, "The development of a simple 3 dimensional facial measuring technique", *Proceeding of Int. Society. Photogram. Symp. on Biostereometrics, Washington DC*, pp. 147, 1974.
9. W. D. Leivesley, "The reliability of contour photography for facial measurements", *British Journal of Orthodontics*, **10**, pp. 34-37, 1983.
10. S. M. Dunn, R. L. Keizer and J. YU, "Measuring the area and volume of the human body with structured light", *IEEE Transactions on System, Man., and Cybernetics*, **19**, pp. 1350-1364, 1989.
11. H. Kaplan, "Structured light finds a home in machine vision", *Photonics at Work, Photonics Spectra, Laura Publishing Co.*, 1994.

12. C. Stanek, N. Shrikhande and G. Hu, "Decision making strategies in contour tracing", *SPIE Imaging Technologies and Applications*, **1778**, pp. 247-257, 1992.

13. H. A. Beyer, T. Kersten and A. Streilein, "Metric accuracy performance of solid-state camera system", *Proceeding of SPIE-Videometrics*, **1820**, pp. 103-110, 1992.

14. R. Y. Tsai, "An efficient and accurate camera calibration techniques for 3-D machine vision", *Proceeding-CVPR'86: IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 364-374, 1986.